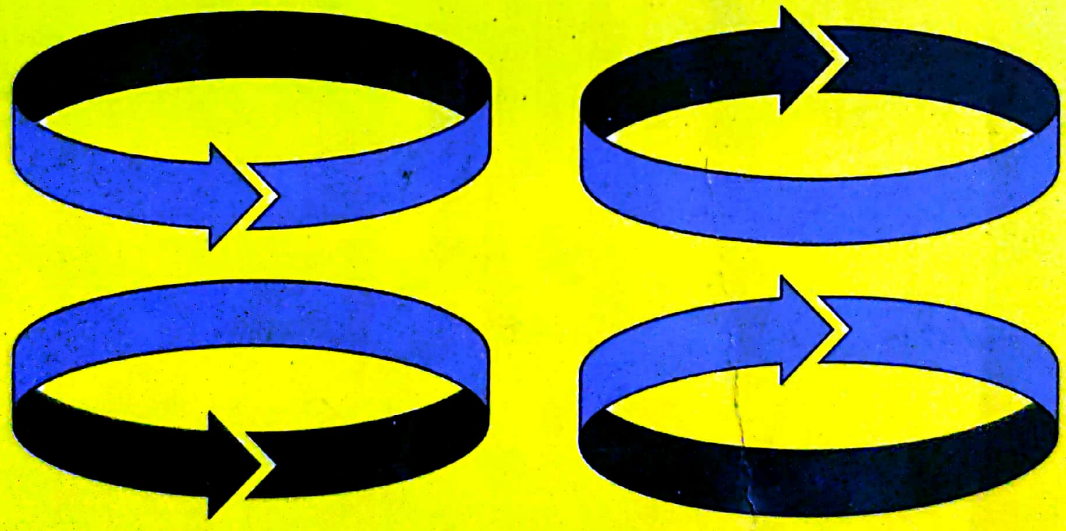
Dr. Ing. ADRIANA DUMITRAȘ

PROIECTAREA REȚELELOR NEURALE ARTIFICIALE



CASA EDITORIALA ODEON
București, 1997.

Scanned with OKEN Scanner

Coperta: ADRIANA DUMITRAȘ

Toate drepturile asupra acestei ediții sunt rezervate
S.C. „NCV ODEON" SRL, str. Th. Aman, nr. 15, sector 1, București
telefax: 01/312.37.56.

ISBN: 973-9008-75-5

Scanned with OKEN Scanner

Mamei mele si Tatalui meu

Scanned with OKEN Scanner

ABSTRACT

The author of the book ("Artificial Neural Network Design") was born on March 24, 1965, in Bucharest. She graduated the Electronics and Telecommunications Department, POLITEHNICA University of Bucharest, Romania, in 1988. In 1996, the author defended her Ph.D. thesis and became a doctor engineer in Electronics. Nowadays, she is with the same department, as Senior assistant professor.

Miss Adriana Dumitraş has been co–author of one book and more than 25 papers, published in scientific journals and proceedings of international conferences. In 1993, she was the recipient of a grant at the Technical University of Denmark, Institute of Circuit Theory and Telecommunications. The author is mainly interested in artificial neural networks and digital signal processing.

The book introduces in its first part (chapters 1 – 4, appendices 1 – 4) main ideas connected with the the artificial neural networks (ANNs), their characteristics as digital signal processing tools and the state of art of neural models and their applications. Theoretical ideas connected with the neural model in general, and with the feedforward ANNs in particular, are presented. They are followed by a synthesis of design methods, based on extensive references (more than 450 titles).

The second part of the book (chapters 5 – 6, appendices 5 – 6) has original contributions regarding the neural design with symmetry constraints and the neural design with robust M estimators as cost functions. The theoretical results are checked experimentally on nonlinear regression applications and compared to those given by standard processing techniques.

The book addresses a large public (prerequisite is elementary mathematics), familiar or not with the field of artificial neural networks. All the new ideas and concepts are explained step by step, so the level is accessible.

All the readers interested in artificial neural models, willing to design and implement their own applications, are welcomed. Both members of the university teaching staff and students will find the book a useful tool for their lectures. At the same time, physicists, mathematicians, engineers, but also biologists, psychologists and physicians will be able to evaluate their own models from the artificial neural network point of view. We also hope that all the readers will be inspired to perform comparisons with their own ideas and sources of inspiration. Looking forward to reading their comments and constructive criticism!

Scanned with OKEN Scanner

CUVÂNT ÎNAINTE

Reţelele neurale artificiale reprezintă obiectul de studiu al domeniului interdisciplinar, aflat la graniţa dintre inginerie, ştiinţele cunoaşterii, ştiinţa calculatoarelor şi ştiinţele fundamentale: matematica, fizica, biologia, etc. Apariţia şi dezvoltarea acestuia a fost rezultatul interferenţei ideilor preluate din domeniile menţionate, precum şi al interesului cercetătorilor cu preocupări diverse. In acest moment, există o comunitate ştiinţifică internaţională bine conturată, cu o activitate remarcabilă în domeniul reţelelor neurale.

In ţara noastră, deşi în ultimii ani au existat preocupări pentru studiul reţelelor neurale artificiale, mai ales în mediul universitar, acest interes a fost susţinut de rare apariţii editoriale. De aceea, în peisajul livresc actual, sperăm că lucrarea de faţă va contribui, la stimularea interesului şi la promovarea rezultatelor cercetării în domeniul menţionat.

Cartea este scrisă din perspectivă inginerească, îmbinând aspectul teoretic şi cel experimental. Obiectivul său este studiul proiectării reţelelor neurale "feedforward" utilizate ca instrumente de prelucrare digitală a semnalelor. Acesta este abordat pe de o parte, din punct de vedere al proiectării topologice (dimensionarea structurii, introducerea constrângerilor de simetrie structurală), precum şi din punct de vedere al proiectării cu funcţii cost diferite de norma L_2 a erorii.

Prima parte a lucrării (cap. 1 – 4, anexele 1 – 4) prezintă locul şi rolul reţelelor neurale artificiale în neuroinformatică, caracteristicile lor ca instrumente de prelucrare digitală a semnalelor şi stadiul actual al dezvoltării modelelor şi aplicaţiilor. Bazele teoretice de abordare a studiului modelului neural şi elemente legate de învăţare, generalizare şi regularizare în reţele "feedforward" sunt incluse în continuare, urmate de o sinteză a metodelor de proiectare a modelelor neurale, documentată pe baza unei bibliografii de peste 450 de lucrări.

Partea a doua a lucrării (cap. 5 – 6, anexele 5 – 6) conţine contribuţii originale legate de proiectarea cu constrângeri de simetrie structurală şi de proiectarea cu funcţii cost estimatori robuşti de tip M. Rezultatele teoretice obţinute sunt verificate experimental în rezolvarea problemelor de regresie neliniară şi comparate cu cele obţinute prin aplicarea tehnicilor de prelucrare clasice.

Cartea se adresează unui public larg, cu cunoştinţe elementare de matematică, familiarizat sau nu, cu domeniul reţelelor neurale artificiale. Toate

Scanned with OKEN Scanner

elementele noi, introduse pe parcurs, sunt explicate, astfel încât nivelul este accesibil.

Lucrarea este utilă tuturor celor interesaţi să studieze modelele neurale artificiale, să proiecteze şi să realizeze propriile aplicaţii, bazate pe structuri neurale. Atât cadrele didactice din învăţământul universitar, cât şi studenţii, vor găsi un instrument de lucru util pentru susţinerea capitolelor de curs. In aceeaşi măsură, cartea se adresează inginerilor, fizicienilor, matematicienilor, informaticienilor, dar şi biologilor, psihologilor şi medicilor, care vor putea regândi abordările specifice, din perspectiva modelelor artificiale prezentate, precum şi tuturor cititorilor, care vor afla posibile repere de comparaţie cu propriile idei, surse de inspiraţie şi reflecţie. Mulţumim anticipat pentru comentariile şi observaţiile acestora, care sunt aşteptate şi binevenite.

Lucrarea a fost redactată de autoare, folosind editorul Microsoft Word 6.0 şi celelalte programe utilitare din pachetul Microsoft MsOffice 4.3, pe un calculator IBM – PC 486 DX2 / 66 MHz.

Realizarea acestei lucrări nu ar fi fost cea prezentă, fără dragostea, sprijinul constant şi înţelegerea Mamei mele şi Tatălui meu. Le mulţumesc şi le dedic lucrarea de faţă, la care au contribuit nu numai afectiv.

Am avut privilegiul să interacţionez constructiv şi distructiv în perioada realizării cărţii cu o serie de persoane, unele dintre acestea personalităţi recunoscute, altele "outsideri" ai domeniului reţelelor neurale artificiale. Tuturor celor care au avut un efect benefic asupra mea personal şi asupra lucrării, în mod direct sau indirect, le sunt în mod sincer şi deplin recunoscătoare. In succesiunea dificilă de urcări şi coborâri de trepte, care a caracterizat studiul realizat, prezenţa lor continuă sau sub formă de intermezzo, a marcat o permanentă reconsiderare a atitudinii mele faţă de tema abordată, faţă de sensul demersului şi faţă de utilitatea rezultatelor obţinute, precum şi menţinerea reperelor fixe de onestitate şi profesionalism.

Ii mulţumesc domnului prof.dr.ing. Vasile Lăzărescu, pentru încrederea necondiţionată pe care mi-a acordat-o, pentru suma lucrurilor pe care le-am învăţat în domeniul profesional, precum şi pentru ajutorul şi feedback-ul critic permanent, legate de toate lucrările realizate până acum, inclusiv cea de faţă.

Scanned with OKEN Scanner

Îi mulţumesc prof. Yu Hen Hu, de la Department of Electrical and Computer Engineering, Universitatea Wisconsin – Madison, Statele Unite, pentru sprijinul său continuu, pentru susţinerea ideii mele de proiectare cu funcţii cost diferite de norma L_2 a erorii, precum şi pentru entuziasmul său molipsitor.

Prof. Lars Kai Hansen, de la Electronics Institute, Universitatea Tehnică din Danemarca, pentru generozitatea cu care mi-a împărtăşit o parte a ideilor sale legate de direcţiile de cercetare în domeniul reţelelor neurale artificiale, pentru sublinierea importanţei problemei de proiectare arhitecturală şi pentru contribuţia sa la saltul calitativ pe care l-am realizat, îi sunt îndatorată.

Îi mulţumesc prof. Hans Gaunholt pentru alegerea mea ca bursier Tempus la Institutul de Teoria Circuitelor şi Telecomunicaţii, Universitatea Tehnică din Danemarca, în perioada februarie – iulie 1993, precum şi pentru întregul sprijin acordat, care mi-a permis să realizez 95% din documentarea pentru această lucrare, precum şi studii preliminare.

Îi sunt recunoscătoare pentru aprecierile asupra lucrării, domnului prof.dr.ing. Victor Neagoe.

Îi sunt recunoscătoare doamnei prof.dr.ing. Rodica Strungaru, pentru înţelegerea şi sprijinul acordat, pentru relaţia profesională creată, caracterizată prin deschidere şi comunicare sinceră.

Le mulţumesc doamnei prof.dr.ing. Tatiana Stănăşilă şi domnului prof.dr.ing. Octavian Stănăşilă, doamnei prof.dr.ing. Gabriela Cone şi domnului prof.dr.ing. Ion M. Popescu, pentru rolul decisiv avut în formarea mea profesională.

Domnului conf.dr.ing. Niculae Tomescu îi datorez sfatul bun şi drept în momente de cumpănă, exemplul de verticalitate a atitudinii.

Prietenilor şi colegilor mei, le mulţumesc pentru discuţiile stimulatoare avute şi pentru susţinerea continuă, precum şi tuturor celor care, fără a fi implicaţi în domeniul reţelelor neurale artificiale, au formulat întrebări utile, regăsite în tonalitatea generală a lucrării.

Bucureşti, Ianuarie 1997 Autoarea

Scanned with OKEN Scanner

CUPRINS

CUVÂNT ÎNAINTE XVI
LISTA DE ABREVIERI ŞI SIMBOLURI XXI
LISTA DE FIGURI XXVIII
LISTA DE TABELE

1. INTRODUCERE 1

1.1 Apariţia modelelor conexioniste 1
1.1.1 Prelucrare paralelă şi distribuită vs. prelucrare secvenţială 1
1.1.2 Inspiraţia din neuroştiinţe 2
1.1.2.1 Sistemul nervos biologic 2
1.1.2.2 Comparaţie între creierul uman şi sistemul de calcul 4
1.1.2.3 Ideile preluate din neuroştiinţe 4
1.2 Domeniul neuroinformaticii 4
1.2.1 Definiţiile modelelor neurale artificiale 5
1.2.2 Terminologie 6
1.2.3 Caracteristicile şi limitele modelelor neurale artificiale 7
1.2.4 Abordări ale reţelelor neurale artificiale şi probleme de studiu 8
1.2.5 Modele neurale artificiale ca instrumente de prelucrare digitală a semnalelor 8
1.2.5.1 Caracteristicile modelelor neurale artificiale ca instrumente de PDS 10
1.2.5.2 Etapele proiectării modelelor neurale artificiale pentru PDS 10
1.2.5.3 Implementări ale modelelor neurale artificiale 11
1.2.5.4 Utilitatea modelelor neurale artificiale ca instrumente de PDS 12
1.2.6 Relaţia dintre neuroinformatică şi neuroştiinţe 12
1.3 Structura lucrării 13

Scanned with OKEN Scanner

2. STADIUL ACTUAL AL DEZVOLTARII STRUCTURILOR ŞI APLICAŢIILOR RETELELOR NEURALE ARTIFICIALE 15

2.1 Trecut (Scurt istoric) 15
2.1.1 Scurt istoric al modelelor neurale artificiale 15
2.1.1.1 Epoca lui Camelot ("Dinozaurii", 1890 – 1960) 15
2.1.1.2 Perioada întunecată (1969 – 1980) 18
2.1.1.3 Renaşterea (1982 – 1986) 19
2.1.2.4 Epoca neoconexionistă (din 1987 până în prezent 19
2.1.2 Scurt istoric al aplicaţiilor modelelor neurale artificiale în prelucrarea digitală a semnalelor 20
2.2 Prezent (Stadiul actual) 22
2.2.1 Stadiul actual al dezvoltării modelelor neurale artificiale 22
2.2.2 Stadiul actual al aplicaţiilor modelelor neurale artificiale în prelucrarea digitală a semnalelor 25
2.3 Viitor (Perspective) 26
2.4 Rezumatul capitolului 31

3. BAZELE TEORETICE ALE ABORDĂRII MODELELOR NEURALE ARTIFICIALE 32

3.1 Caracteristicile modelelor neurale artificiale 32
3.1.1 Caracteristicile nodului (perceptron simplu). Micro–structura neurală 33
3.1.1.1 Modele de perceptron simplu 33
■ Modele de perceptron simplu fără reactie 33
■ Modelul de perceptron simplu cu reacţie 36
3.1.1.2 Funcţii de activare 38
3.1.2 Topologia reţelei. Mezo – structura şi macro–structura neurală 40
3.1.2.1 Mezo – structura neurală 40
■ Mezo – structura neurală "feedforward" 44
■ Mezo – structura neurală "feedback" 45
■ Relaţia dintre mezo – structura "feedforward" şi cea "feedback" 46
3.1.2.2 Macro – structura neurală 47
3.1.3 Modelul datelor şi criteriul de cost 47
3.1.4 Legea de învăţare 48

Scanned with OKEN Scanner

3.1.4.1 Definiţiile învăţării 48
3.1.4.2 Legi de învăţare 49
3.1.5 Condiţii de stabilitate 51
3.1.6 Clasificarea modelelor neurale artificiale 52
3.2 Structuri neurale "feedforward" 52
3.2.1 Notaţii 52
3.2.2 Învăţarea în structurile neurale "feedforward" 54
3.2.2.1 Învăţarea în structurile "feedforward" ca proces de aproximare 54
■ Problema de aproximare 54
■ Problema de reprezentare. Aproximare universală 55
☐ Teorema Kolmogorov 55
☐ Teorema Kolmogorov – Sprecher 56
■ Eroare de învăţare şi eroare de test. Fenomenul de supraînvăţare 57
■ Compromisul deplasament – dispersie 58
3.2.2.2 Învăţarea în structurile "feedforward" ca proces de optimizare 59
3.2.2.3 Algoritmi de învăţare în structurile "feedforward" 61
■ Determinarea direcţiei de căutare a minimului în spaţiul parametrilor 61
■ Metode deterministe de ordinul întâi 62
☐ Metoda gradientului negativ 62
■ Metode deterministe de ordinul doi 62
☐ Metoda gradientului conjugat 63
☐ Metoda Newton 64
☐ Metode cvasi – Newton 65
☐ Metoda pseudo – Newton 66
■ Metode stohastice 67
■ Concluzii privind metodele de căutare prezentate 67
3.2.3 Generalizarea în structurile neurale "feedforward" 68
3.2.3.1 Evaluarea capacităţii de generalizare 69
3.2.3.2 Imbunătăţirea capacităţii de generalizare 72
■ Estimarea capacităţii de generalizare a reţelei în timpul învăţării şi oprirea învăţării când aceasta începe să descrească 72
■ Limitarea numărului de parametri ai reţelei 73
■ Alegerea numărului de patternuri de învăţare 73
3.2.4 Reducerea complexităţii modelului. Regularizare 74

Scanned with OKEN Scanner

3.2.4.1 Principiul regularizării 75
3.2.4.2 Soluţia ecuaţiei de regularizare 76
3.2.4.3 Regularizare cu scăderea ponderilor 78
3.3 Rezumatul capitolului 79

4. METODE DE PROIECTARE A STRUCTURILOR NEURALE "FEEDFORWARD" 81

4.1 Alegerea modelului neural. Criterii de optimalitate 81
4.1.1 Criterii de evaluare a modelului 81
4.1.2 Argumente în favoarea reducerii complexităţii modelului 82
4.1.3 Avantajele şi dezavantajele structurilor neurale "feedforward" de dimensiune redusă 83
4.1.4 Criterii de optimalitate 84
4.2 Obiectivul şi etapele proiectării structurilor "feedforward" 85
4.2.1 Alegerea numărului de straturi 85
4.2.2 Alegerea numărului de perceptroni simpli din fiecare strat 86
4.2.3 Alegerea numărului de conexiuni 87
4.2.4 Proiectarea reţelelor neurale ontogenice 87
4.3 Metode empirice pentru proiectarea structurilor neurale "feedforward" 87
4.3.1 Metode ad hoc 88
■ Relaţia Chauvin 88
■ Relaţia Kudrycki 88
■ Relaţia Hecht – Nielsen 88
■ Relaţia Baum – Haussler – Widrow 88
4.3.2 Metode bazate pe stabilirea tipului şi numărului zonelor de decizie realizate de reţeaua neurală în spaţiul multidimensional 89
■ Metoda (I) 89
■ Metoda Lippmann (II) 90
4.4 Metode bazate pe criterii statistice 91
4.4.1 Formularea problemei 91
4.4.2 Forma generală a criteriilor statistice pentru proiectarea structurilor neurale "feedforward" 92
4.4.3 Criterii statistice pentru proiectarea structurilor neurale "feedforward" 93
■ Criteriul FPE ("Final Prediction Error") (I) 94

Scanned with OKEN Scanner

- Criteriul AIC ("Akaike's Information Theoretic Criterion") (II) 94
- Criteriu statistic (III) 95
- Criteriul FPER ("Final Prediction Error criterion for Regularized networks") (IV) 95
- Criteriul NIC ("Network Information Criterion") (V) 96
- Criteriul PMDL ("Predictive Minimum Description Length") (VI) 101

4.4.4 Comparaţie a criteriilor statistice considerate pentru proiectarea structurilor neurale "feedforward" 103

4.5 Metode ontogenice constructive 106

4.5.1 Formularea problemei 107

4.5.1.1 Spaţiul de căutare 107

4.5.1.2 Starea iniţială şi starea finală 107

4.5.1.3 Strategia de căutare 107

4.5.2 Clasificarea metodelor ontogenice constructive 108

4.5.3 Metode ad hoc 110

- Metodă ad hoc (I) 110
- Metodă ad hoc (II) 110
- Metoda G ("Generation") (III) 110
- Metoda "Tiling" (IV) 110
- Metoda Lee – Tenorio (V) 111

4.5.4 Metode de partiţionare a spaţiului de intrare 111

- Metoda RCE ("Restricted Coulomb Energy") (I) 112
- Metoda GAL ("Grow and Learn") (II) 112
- Algoritmul OSA ("Oilspot Algorithm") (III) 113

4.5.5 Metode de corectare a erorii 113

4.5.5.1 Metode ce interconectează total elementele introduse în reţea, cu cele existente 113

- Metoda CASCOR ("Cascade Correlation") 113
- Metoda CL ("Constructive Learning") (II) 122
- Algoritmul LPPA ("Linear – Programming Perceptron Algorithm") (III) 123

4.5.5.2 Metode ce interconectează parţial elementele introduse în reţea, cu cele existente 123

- Metoda GMDH ("Group Method of Data Handling") 123

4.5.6 Metode care generează arbori 124

- Algoritmul Upstart (I) 124
- Metodă care generează un arbore (II) 125

Scanned with OKEN Scanner

4.5.7 Metode care construiesc reteaua pornind de la un arbore 128
■ Metoda retelelor entropice (I) 128
■ Metodă care construieste reteaua pornind de la un arbore (II) 131
4.5.8 Metode de modularizare a retelei 132
■ Metodă de modularizare a retelei (I) 132
■ Metodă de modularizare a retelei (II) 132
■ Metoda TADCM ("Task Decomposition through Competition in a Modular connectionist architecture (III) 132
■ Metoda TACOMA ("Task Decomposition by Correlation Measures") (IV) 133
4.5.9 Alte metode constructive pentru structuri neurale "feedforward" 134
4.5.10 Comparaţie între metodele ontogenice constructive pentru proiectarea structurilor "feedforward" 137
4.6 Metode ontogenice "distructive" 142
4.6.1 Formularea problemei 142
4.6.2 Clasificarea metodelor ontogenice "distructive" 144
4.6.3 Metode ontogenice "distructive" care elimină ponderi în exces 145
4.6.3.1 Metode drastice 145
4.6.3.2 Metode de grupare a ponderilor 146
■ Metodă de tip "local bottleneck" (I) 146
■ Metodă de tip "distributed bottleneck") (II) 145
■ Metode de tip "soft weight sharing" (III) 148
4.6.3.3 Metode bazate pe estimarea senzitivităţii 149
■ Metoda OBD ("Optimal Brain Damage") (I) 149
■ Metoda OCD ("Optimal Cell Damage") (II) 156
■ Metoda OBS ("Optimal Brain Surgeon") (III) 157
■ Metoda SSM ("Statistical Stepwise") (IV) 158
■ Metodă de calcul a senzitivităţii (V) 160
4.6.3.4 Metode "distructive" bazate pe regularizare 162
■ Metodă "distructivă" bazată pe regularizare de tip "weight decay" şi distribuţie apriori Laplace a ponderilor (I) 162
■ Metodă "distructivă" bazată pe regularizare de tip "weight decay" şi distribuţie apriori Gauss a ponderilor (II) 162
■ Metodă bazată pe regularizare (III) 163
■ Metodă bazată pe regularizare (IV) 164

Scanned with OKEN Scanner

■ Metoda OBD cu regularizare de tip "weight decay" (V) 165
■ Metoda OBS cu regularizare de tip "weight decay" (VI) 166
■ Metodă bazată pe regularizare (VII) 167
■ Metodă bazată pe regularizare (VIII) 168
■ Metodă bazată pe regularizare (IX) 169
■ Metodă bazată pe regularizare (X) 170
■ Metoda CSDF ("Convergence Supression and Divergence Facilitation") (XI) 171
4.6.4 Metode "distructive" de eliminare a nodurilor ascunse în exces 171
4.6.4.1 Metode drastice 171
4.6.4.2 Metode de grupare a ponderilor 171
■ Metodă de skeletonizare (I) 171
■ Metoda CSDF ("Convergence Supression and Divergence Facilitation") (II) 172
4.6.4.3 Metode bazate pe calculul senzitivităţii 171
■ Metodă de skeletonizare (I) 173
■ Metodă de câştigare a competiţiei (II) 175
4.6.4.4 Metode bazate pe regularizare 175
4.6.4.5 Alte metode de eliminare a câte unui nod ascuns 175
■ Metodă bazată pe corelaţie (I) 175
■ Metodă de aproximare a normei Frobenius (II) 177
4.6.5 Comparaţie a metodelor ontogenice "distructive" de proiectare a structurilor neurale "feedforward" 177
4.7 Metode ontogenice mixte 183
4.7.1 Clasificarea metodelor ontogenice mixte 183
4.7.2 Metode secvenţiale (constructive – "distructive") 183
■ Metodă secvenţială (I) 183
■ Metodă secvenţială (II) 183
■ Metodă secvenţială (III) 184
■ Metodă secvenţială (IV) 184
■ Metodă secvenţială (V) 184
□ Etapa constructivă, folosind algoritmul CASCOR 184
□ Etapa de reducere a structurii, folosind algoritmul OBD 184
■ Metoda G – S ("Generating–Shrinking") (VI) 192
■ Metodă secvenţială (VII) 196
4.7.3 Metode nesecvenţiale 196

Scanned with OKEN Scanner

■ Metoda CASCOR controlată (I) 196
■ Metoda DNAL ("Dynamic Node Architecture Learning") (II) 198
4.8 Comparaţie între metodele de proiectare a structurilor neurale "feedforward" 200
4.9 Rezumatul capitolului 202

5. PROIECTAREA STRUCTURILOR NEURALE "FEEDFORWARD" CU CONSTRÂNGERI DE SIMETRIE 203

5.1 Necesitatea impunerii constrângerilor de simetrie în proiectarea reţelelor neurale "feedforward" 203
5.2 Formularea problemei de proiectare neurală cu constrângeri de simetrie 204
5.3 Metoda MOBD ("Modified Optimal Brain Damage") 205
5.4 Metoda SVD–MOBD ("Modified Optimal Brain Damage with Singular Value Decomposition") 207
5.4.1 Utilizarea transformărilor pentru proiectarea RNA 207
5.4.2 Descompunerea SVD 208
5.4.3 Utilizarea descompunerii SVD în metoda MOBD 209
5.5 Metoda TOBD ("Tridiagonal Optimal Brain Damage") 210
5.5.1 Matrici rare 210
5.5.2 Argumente în favoarea obţinerii matricilor tridiagonale de ponderi pentru structuri neurale simetrice 212
5.5.3 Transformarea de reflexie 213
5.5.4 Utilizarea transformării de reflexie în metoda TOBD 214
5.6 Rezultate experimentale ale aplicării metodelor propuse, de proiectare cu constrângeri de simetrie 218
5.6.1 Problema de studiu 218
5.6.1.1 Setul datelor 218
5.6.1.2 Modelul neural 219
■ Structura reţelei neurale 219
■ Algoritmul de învăţare 219
5.6.2 Evaluarea performanţelor 220
■ Eroarea de învăţare, eroarea de test 220
■ Raportul semnal – zgomot la vârf 221
■ Valorile ponderilor obţinute 221
■ Numărul total al parametrilor structurii neurale obţinute 222
■ Numărul operaţiilor de "înmulţire + adunare" efectuate în etapa de test 222

Scanned with OKEN Scanner

■ Timpul necesar pentru etapa de test 222
■ Spaţiul necesar implementării 223
5.6.3 Etapa de învătare 224
5.6.4 Performanţele reţelei neurale "feedforward" total interconectată, cu algoritmul BKP standard 225
5.6.5 Performanţele reţelei neurale "feedforward" simetrică, parţial interconectată, rezultată în urma aplicării metodei MOBD 225
5.6.6 Performanţele reţelei neurale "feedforward" simetrică, parţial interconectată, rezultată în urma aplicării metodei SVD–MOBD 226
5.6.7 Performanţele reţelei neurale "feedforward" simetrică, parţial interconectată, cu matrice de ponderi tridiagonală, rezultată în urma aplicării metodei TOBD 227
5.6.8 Performanţele reţelei neurale "feedforward" parţial interconectată, rezultată urma aplicării metodei OBD 228
5.6.9 Interpretarea comparativă a rezultatelor experimentale obţinute prin aplicarea metodelor propuse 242
5.7 Concluzii, direcţii de studiu şi perspective 250
5.7.1 Concluzii teoretice 250
5.7.2 Concluzii experimentale 250
5.7.3 Direcţii de studiu şi perspective 252
5.8 Rezumatul capitolului 253

6. PROIECTAREA STRUCTURILOR NEURALE "FEEDFORWARD" CU FUNCŢII COST DIFERITE DE NORMA L_2 254

6.1 Alegerea criteriului de cost în funcţie de distribuţia erorilor în setul de date 255
6.1.1 Alegerea funcţiei cost când funcţia densitate de probabilitate a zgomotului este cunoscută 255
6.1.1.1 Distribuţie gaussiană a erorilor 255
6.1.1.2 Distribuţie dublu exponenţială a erorilor 255
6.1.1.3 Distribuţie uniformă a erorilor 255
6.1.1.4 Distribuţie gaussiană generalizată a erorilor 256
6.1.1.5 Distribuţie Cauchy a erorilor 256
6.1.2 Alegerea funcţiei cost când funcţia densitate de probabilitate a zgomotului nu este cunoscută 256
6.1.2.1 Utilizarea estimatorilor de tip M 257
6.1.2.2 Utilizarea estimatorilor de tip M modificaţi

Scanned with OKEN Scanner

şi variantelor acestora 257
■ Alegerea unei funcţii cost compuse 257
■ Funcţie Φ cu formă adaptivă 258
■ Funcţia cost Lister 258
6.1.2.3 Utilizarea funcţiilor ITF ("Information Theoretic Functions") 259
■ Entropia ca funcţie cost 259
■ Funcţia cost Kullback – Liebler 259
■ Informaţia mutuală (transinformaţia) ca funcţie cost 261
6.1.2.5 Alte funcţii cost 261
6.2 Alegerea funcţiei cost pentru o aplicaţie dată 262
6.2.1 Funcţii rezonabile 262
6.2.2 Funcţii care nu sunt rezonabile 263
6.2.2.1 Funcţiile cost CFM ("Classification Figures of Merit") 263
6.2.2.2 Funcţia cost Minkowski (norma L_R) 263
6.2.2.3 Funcţia cost cu valori nebinare ale ieşirilor dorite 264
6.2.3 Alte funcţii cost 264
6.3 Proiectarea reţelelor "feedforward" cu funcţii cost estimatori robuşti de tip M 264
6.3.1 Problema de proiectare ca optimizare mono – criteriu 265
6.3.1.1 Formularea problemei de optimizare mono – criteriu 265
6.3.1.2 Modificarea valorilor parametrilor în algoritmul BKP, varianta standard 266
6.3.1.3 Modificarea valorilor parametrilor în algoritmul BKP, pentru funcţii cost estimatori robuşti de tip M 268
6.3.1.4 Modificarea valorilor parametrilor în algoritmul BKP, pentru funcţii cost estimatori robuşti de tip M şi pantă variabilă a funcţiei de activare a perceptronului simplu 268
6.3.2 Problema de proiectare ca optimizare multi – criteriu 269
6.3.2.1 Formularea problemei de optimizare multi – criteriu 270
6.3.2.2 Modificarea valorilor parametrilor în algoritmul BKP, pentru funcţii cost estimatori robuşti de tip M modificaţi 272
6.3.2.3 Modificarea valorilor parametrilor în algoritmul BKP, pentru funcţii cost estimatori robuşti de

Scanned with OKEN Scanner

tip M modificați și pantă variabilă a funcției de activare a perceptronului simplu 273
6.4 Rezultate experimentale ale proiectării cu funcții cost estimatori robuști de tip M 274
6.4.1 Problema de studiu 274
6.4.1.1 Setul datelor 275
6.4.1.2 Modelul neural 275
■ Structura rețelei neurale 275
■ Algoritmul de învățare 277
6.4.2 Evaluarea performanțelor 277
6.4.3 Performanțele rețelelor neurale "feedforward" cu funcții cost estimatori robuști de tip M, în varianta de optimizare mono – criteriu 277
6.4.3.1 Cazul pantei constante a funcției de activare 277
■ Experimentul 1 278
■ Experimentul 2 278
■ Experimentul 3 279
■ Experimentul 4 279
■ Experimentul 5 280
6.4.3.2 Cazul pantei variabile a funcției de activare 283
■ Experimentul 6 284
■ Experimentul 7 285
■ Experimentul 8 286
■ Experimentul 9 286
6.4.4 Performanțele rețelelor neurale "feedforward" cu funcții cost estimatori robuști de tip M, în varianta de optimizare multi – criteriu 286
6.4.4.1 Cazul pantei constante a funcției de activare 286
6.4.5 Interpretarea comparativă a rezultatelor experimentale obținute cu funcții cost estimatori robuști de tip M 288
6.5 Concluzii, direcții de studiu și perspective 292
6.5.1 Concluzii teoretice 292
6.5.2 Concluzii experimentale 292
6.5.3 Direcții de studiu și perspective 294
6.6 Rezumatul capitolului 294

7. OBSERVAȚII FINALE, CONCLUZII ȘI PERSPECTIVE 295

7.1 Observații finale și concluzii 295
7.2 Direcții de studiu și perspective 296
7.5 In loc de încheiere 296

Scanned with OKEN Scanner

ANEXA 1 SUMAR AL PROGRAMELOR UNIVERSITARE ȘI INDUSTRIALE DE CERCETARE ÎN DOMENIUL REȚELELOR NEURALE ARTIFICIALE (1990–1996) 297

ANEXA 2 SUMAR AL IMPLEMENTĂRILOR REȚELELOR NEURALE ARTIFICIALE ÎN MEDIUL UNIVERSITAR ȘI INDUSTRIAL (1990–1996) 306

ANEXA 3 ALTE MODELE DE PERCEPTRON SIMPLU 313

ANEXA 4 FUNCȚII DE ACTIVARE ALE PERCEPTRONULUI SIMPLU 323

ANEXA 5 REZULTATELE STUDIULUI PRIVIND MODIFICAREA ADAPTIVĂ A PANTEI FUNCȚIEI DE ACTIVARE 332

ANEXA 6 INTERPRETAREA ÎN PLANUL COMPLEX A REZULTATELOR ÎNVĂȚĂRII REȚELEI NEURALE "FEEDFORWARD" 339

BIBLIOGRAFIE 345

ABSTRACT 372

Scanned with OKEN Scanner

LISTA DE ABREVIERI ȘI SIMBOLURI

❶ **In textul lucrării au fost utilizate prescurtări** din limba română (exemplu: RNA = rețele neurale artificiale), dar s-au păstrat abrevierile standard din limba engleză (exemplu: PDP = prelucrarea paralelă și distribuită, ML = "Maximum Likelihood" = verosimilitate maximă, etc.)

❷ **S-au notat cu:**
Literă mică, **bold** = vector (exemplu **x** = vectorul x)
Literă mare, **bold** = matrice (exemplu **X** = matricea X)

❸ Simbolurile care apar în textul cursiv sunt italice, cu excepția literelor **w**, **v**, **W**, care ar putea genera confuzii.

ABREVIERI

AIC	criteriul statistic Akaike ("Akaike's Information Criterion")
BKP	algoritmul de propagare inversă ("Backpropagation Algorithm")
det	determinant
EPM	eroare pătratică medie
EPMN	eroare pătratică medie normată
exp	funcția exponențială
FFWD	"feedforward" (rețea neurală)
log	funcția logaritm
Mn	monotonicitatea funcției
ML	metoda verosimilității maxime ("Maximum Likelihood")
Nm	nemonotonicitatea funcției
NrReg	numărul regiunilor separabile liniar în spațiul de intrare al rețelei neurale artificiale
op_{inv}	numărul operațiilor "înmulțire + adunare" efectuate în etapa de învățare
op_{test}	numărul operațiilor "înmulțire + adunare" efectuate în etapa de test
$O(\bullet)$	ordinul
PDP	prelucrare paralelă și distribuită
PSNR	raport semnal – zgomot la vârf
RBF	rețea cu bază radială ("Radial Basis Function Network")

Scanned with OKEN Scanner

RCC	rata de clasificare corectă
RD	raportul determinanţilor
RNA	reţele neurale artificiale
RPM	reţea neurală perceptron multistrat
sign	funcţia semn
tanh	funcţia tangentă hiperbolică
tr	operatorul "urma matricii"

PRINCIPALELE SIMBOLURI

$\mathbf{A}$	matricea hessiană a funcţiei cost regularizată $\boldsymbol{A} = \boldsymbol{H} + \boldsymbol{R}$
$\hat{\mathbf{A}}$	matrice care aproximează matricea $\boldsymbol{A}$
$\mathbf{A}^{-1}$	inversa matricii $\boldsymbol{A}$
$\mathbf{A}^{+}$	matricea pseudoinversă Moore – Penrose a matricii $\boldsymbol{A}$
α	moment
b_j	pragul ("bias") perceptronului simplu j
$\binom{u}{m}$	coeficient al dezvoltării binomiale
C	funcţia cost
C_{inv}	eroarea de învăţare (valoarea funcţiei cost după etapa de învăţare)
C_r	eroarea de regularizare; se mai notează cu $C_{complexitate}$
C_g	eroarea de generalizare
C_j	clasa j, într-o problemă de clasificare
$\mathbf{C}$	setul tuturor conexiunilor din reţea
d	valoare dorită (componentă a vectorului $\boldsymbol{d}$)
$\mathbf{d}$	vectorul de ieşire dorit: $\mathbf{d} = [d_1, d_2, ..., d_N]$
$\mathbf{d}(\xi)$	vectorul de ieşire dorit, pentru patternul de intrare ξ $\mathbf{d}(\xi) = [d_1(\xi), d_2(\xi), ..., d_N(\xi)]$
$\mathbf{d}$	direcţia de căutare a minimului în spaţiul parametrilor
δ	① δ – independenţă liniară (metoda Wang); ② $\delta\mathbf{w}$ este o mică variaţie a vectorului de ponderi $\mathbf{w}$
∂	operatorul de derivare
D	funcţia discrepanţă, în criteriul NIC
$\mathbf{D}$	matricea valorilor dorite de ieşire pentru cei N neuroni de ieşire; $\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, ..., \mathbf{d}_P]^T$
$\mathcal{D}$	setul de ponderi care urmează să fie şterse (metoda OBD)
Δ	① întârzierea de o unitate de timp; ② setul datelor de învăţare
$E\{\bullet\}$	operator de mediere statistică

Scanned with OKEN Scanner

e_j	eroarea în nodul de ieşire j
$e(w; \xi)$	vectorul erorilor de ieşire, pentru patternul de intrare ξ
f	funcţia de activare a perceptronului simplu
f′	derivata de ordinul întâi a funcţiei de activare a perceptronului simplu
F	funcţia globală ce trebuie aproximată
F_w	funcţia intrare – ieşire implementată de reţeaua neurală "feedforward"; prescurtare pentru $F(w; x)$
Φ	operator liniar (pseudo) diferenţial
g	proiecţie unu la unu (în definiţia 3.3)
g_j	① funcţie reală şi continuă; ② constrângeri (par. 3.2)
G	① funcţie liniară de transfer (par. 3.1); ② funcţie Green
$\mathbf{G}$	eroarea de generalizare a modelului estimat
Γ	eroarea medie de generalizare (media valorilor $\mathbf{G}$ pe seturile de date)
$\hat{\Gamma}$	estimatorul erorii medii de generalizare
h	indice pentru elementele stratului ascuns (perceptroni simpli din stratul ascuns, componentele vectorului de ieşire din stratul ascuns; $1 \le h \le H$
h_r	funcţii Hermite
H	numărul perceptronilor simpli din stratul ascuns (dacă există mai multe straturi ascunse, atunci numărul elementelor fiecăruia se notează cu H_1, H_2, etc.)
$\mathcal{H}$	entropia
H	matrice hessiană
χ	① parametru de regularizare; ② funcţie reală şi continuă (par. 3.2)
i	indice pentru elementele de intrare (perceptroni simpli din stratul de intrare, componentele vectorului de intrare; $1 \le i \le M$
I^M	cubul M – dimensional; $I^M = [0, 1]^M$
I_S	informaţia lui Shannon
I	matricea unitate
j	indice pentru elementele stratului de ieşire (perceptroni simpli din stratul de ieşire, componentele vectorului de ieşire obţinut, componentele vectorului de ieşire dorit; $1 \le j \le N$
J	matrice iacobiană
k	① indexul discret al timpului; ② indexul numărului de epoci
K	numărul total de epoci
L	① funcţia Lagrange; ② numărul straturilor reţelei "feedforward"
L_2	funcţia cost eroare pătratică medie
λ_i	① multiplicatorii Lagrange; ② valoarea proprie indice i
M	numărul perceptronilor simpli din stratul de intrare
$\mathbf{M}$	modelul neural statistic
μ	① valoarea medie a distribuţiei; ② constantă mică pozitivă
net	suma ponderată a intrărilor perceptronului simplu; $net = w^T x$

Scanned with OKEN Scanner

N	numărul perceptronilor simpli din stratul de ieşire
η	rata de învăţare
η_k	rata de învăţare la iteraţia / epoca k
∇	operatorul gradient
∇^2	operatorul pentru calculul matricii hessiene a argumentului
Ω^{Π}	subset nevid al spaţiului euclidian Π – dimensional
p	funcţia densitate de probabilitate
P	numărul patternurilor din setul de învăţare
ψ	funcţie continuă reală monoton crescătoare
Π	numărul total de parametri ai structurii
Π_{ef}	numărul efectiv de parametri ai structurii
$\prod$	operatorul produs
Q	① numărul punctelor grilei (în definiţia 3.3); ② funcţie (metoda Wang)
$\mathbf{R}$	matrice de regularizare
$\Re$	mulţimea numerelor reale
$\Re^{\Pi}$	spaţiul parametrilor reţelei
s	panta funcţiei de activare a perceptronului simplu
s_j	salienţa vectorului de ponderi $\mathbf{w}_j$
S_h	corelaţia dintre ieşirea noului nod introdus şi semnalul eroare în nodul de ieşire j (algoritmul CASCOR)
$\boldsymbol{S}$	set de puncte (în definiţia 3.3)
σ	deviaţia standard
σ^2	dispersia
$\sum$	operatorul sumă
t	indicele timpului continuu
T	① numărul patternurilor din setul de test; ② în poziţia superscript indică operaţia de transpunere a vectorului / matricii
$\mathbf{u}_i$	vectorul unitate, cu direcţia $\mathbf{w}_i(\xi)$, pentru patternul de intrare ξ
v_{hj}	ponderea conexiunii dintre nodul h din stratul ascuns şi nodul j din stratul de ieşire
$\mathbf{v}$	vectorul ieşirilor din primul strat ascuns (metoda Wang)
$\mathbf{V}$	matricea tuturor vectorilor $\mathbf{v}$ (metoda Wang)
w_q	componenta indice q a vectorului de ponderi Π – dimensional
w_{ij}	ponderea conexiunii dintre perceptronii simpli (oarecare) i şi j
w_{ih}	ponderea conexiunii dintre nodul i din stratul de intrare şi nodul h din stratul ascuns
$\mathbf{w}, \mathbf{w}_{\Pi}$	setul total de ponderi (atunci când nu este necesară utilizarea notaţiei separate $\mathbf{w}$, $\mathbf{v}$); $\mathbf{w} = [w_1, ..., w_q, ..., w_{\Pi}]^T$
$\mathbf{w}_0$	① vectorul iniţial al ponderilor; ② punct de extrem fără constrângeri al funcţiei cost

Scanned with OKEN Scanner

$\mathbf{w}^*$	vectorul optimal de ponderi
$\mathbf{w}_j$	vectorul ponderilor conexiunilor care intră în perceptronul simplu j
$\mathbf{w}_k$	vectorul de ponderi la pasul (iterația / epoca) k
$\hat{\mathbf{w}}$	estimatorul vectorului de ponderi $\mathbf{w}$
$\mathbf{W}$	matricea de ponderi, ce are drept coloane vectorii $\mathbf{w}_j$
$\mathbf{x}$	vectorul de intrare; $\mathbf{x} = [x_1, x_2, ..., x_M]$
$\mathbf{x}(\xi)$	vectorul de intrare, pentru patternul ξ; $\mathbf{x}(\xi) = [x_1(\xi), x_2(\xi),...., x_M(\xi)]$
$\mathbf{X}$	matricea (setul) de vectori de intrare; $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ... , \mathbf{x}_P]^t$
ξ	indexul patternului; $1 \le \xi \le P$
y	valoarea ieşirii perceptronului simplu
y_j	valoarea ieşirii perceptronului simplu j
$\mathbf{y}$	vectorul de ieşire obţinut; $\mathbf{y} = [y_1, y_2, ..., y_N]$
$\mathbf{y}(\xi)$	vectorul de ieşire obţinut, pentru patternul de intrare ξ; $\mathbf{y}(\xi) = [y_1(\xi), y_2(\xi), ..., y_N(\xi)]$
$\mathbf{Y}$	matricea valorilor de ieşire ale celor N neuroni de ieşire, pentru toate patternurile de învăţare; $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, ... , \mathbf{y}_P]^T$
$\mathbf{z}$	vectorul de ieşire al nodurilor din stratul ascuns; $\mathbf{z} = [z_1, z_2,..., z_H]$
$\mathbf{z}(\xi)$	vectorul de ieşire al nodurilor din stratul ascuns, pentru patternul de intrare ξ; $\mathbf{z}(\xi) = [z_1(\xi), z_2(\xi), ..., z_H(\xi)]$
Z_{inv}, Z_r	constante de normalizare
$\mathbf{Z}$	matricea valorilor de ieşire ale celor H neuroni ascunşi, pentru toate patternurile de învăţare; $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, ... , \mathbf{z}_P]^T$
$\in$	simbolul de apartenenţă
$\langle \bullet \rangle$	operatorul produs scalar
$\angle$	unghiul dintre vectorii argument
$\cong$	aproximativ egal

Scanned with OKEN Scanner

LISTA DE FIGURI

Figura 1.1	Principalele capitole de studiu în domeniul reţelelor neurale	9
Figura 1.2	Dezvoltarea unui sistem de prelucrare neural	11
Figura 1.3	Structura lucrării	13
Figura 2.1	Stadiul actual al dezvoltării modelelor neurale artificiale	23
Figura 2.2	Tipuri de implementări ale modelelor neurale artificiale	25
Figura 2.3	Sistemele de calcul din generaţia a cincea	28
Figura 2.4	Nivelele de inteligenţă artificială în sistemele de calcul din generaţia a şasea	28
Figura 3.1	Tipuri de reţele neurale	32
Figura 3.2	Model general de perceptron simplu fără reacţie	34
Figura 3.3	Modelul perceptronului simplu McCulloch – Pitts	34
Figura 3.4	Modelul perceptronului simplu cu reacţie la nivelul activării locale	37
Figura 3.5	Modelul perceptronului simplu cu reacţie locală la nivelul sinapsei	37
Figura 3.6	Modelul perceptronului simplu cu reacţie locală la ieşire	37
Figura 3.7	Modelul perceptronului simplu GLRGF ("General Locally Recurrent Generally Feedforward")	38
Figura 3.8	Clasificarea mezo–structurilor neurale	41
Figura 3.9	Mezo – structura neurală "feedforward"	44
Figura 3.10	Mezo – structura neurală "feedback"	46
Figura 3.11	Invăţare supervizată	50
Figura 3.12	Invăţare semisupervizată	50
Figura 3.13	Invăţare nesupervizată	50
Figura 3.14	Curbe tipice de eroare	59
Figura 3.15	Eroarea de învăţare şi eroarea de test, în funcţie de numărul nodurilor ascunse	74
Figura 4.1	Relaţia dintre învăţare, generalizare şi complexitatea modelului neural	84

Scanned with OKEN Scanner

Figura 4.2	Complexitatea $C(H)$ în funcție de numărul nodurilor ascunse (reprezentare pentru H dat)	105
Figura 4.3	Dimensiunea rețelei selectate (numărul nodurilor ascunse) prin aplicarea criteriilor statistice	105
Figura 4.4	Construirea structurii prin algoritmul "Tiling"	111
Figura 4.5	Structura inițială a rețelei CASCOR	114
Figura 4.6	Structura rețelei CASCOR după adăugarea unui nod ascuns	115
Figura 4.7	Structura rețelei CASCOR după adăugarea a două noduri ascunse	117
Figura 4.8	Structura rețelei CASCOR, cu notațiile utilizate	119
Figura 4.9	Rețea GMDH	124
Figura 4.10	Construirea structurii neurale cu algoritmul Upstart	125
Figura 4.11	Arborele generat în cazul aplicării metodei pentru aproximarea funcției $x^5 + y^5$	128
Figura 4.12	Varianta analogică a problemei SAU – EXCLUSIV	130
Figura 4.13	Arborele generat pentru problema SAU – EXCLUSIV	130
Figura 4.14	Rețeaua entropică rezultată prin translația arborelui din fig. 4.13 în rețea	130
Figura 4.15	Organigrama testelor efectuate în pasul ❺ al algoritmului	136
Figura 4.16	După învățare, se presupune că eroarea de învățare a rețelei a atins un punct de minim local	151
Figura 4.17	Câmpul de suprimare a convergenței și de facilitare a divergenței	172
Figura 4.18	Problema de studiu: refacerea imaginii folosind rețeaua perceptron multistrat	186
Figura 4.19	Experimentele 1 și 2, pentru cazul structurii fixe, în scopul alegerii dimensiunii stratului de ieșire al rețelei	187
Figura 4.20	Structura inițială și structura finală în pasul ❶ (CASCOR)	189
Figura 4.21	Organigrama etapei de reducere a rețelei, în metoda G – S	195
Figura 5.1	Semnalul "chirp" de învățare și semnalul "chirp" de test	219
Figura 5.2	Corelația dintre semnalul "chirp" de intrare și cel de ieșire dorit	219
Figura 5.3	Semnalul de ieșire obținut, semnalul de ieșire dorit și eroarea pătratică medie, în etapa de test	220
Figura 5.4	Semnalul de ieșire obținut și semnalul de ieșire dorit (eșantioane cu index în intervalul 1500 – 1575)	229
Figura 5.5	Semnalul de ieșire obținut, semnalul de ieșire dorit și	

Scanned with OKEN Scanner

	eroarea pătratică medie în etapa de test, după aplicarea metodei MOBD	230
Figura 5.6	Semnalul de ieşire obţinut după aplicarea metodei MOBD şi semnalul de ieşire dorit (eşantioane cu index în intervalul 1500 – 1575)	230
Figura 5.7	Reînvăţare după etapele de eliminare a ponderilor, în metoda MOBD. Vârfurile curbei indică eliminarea conexiunilor	230
Figura 5.8	Ponderile şi salienţele ponderilor după aplicarea metodei MOBD pentru conexiunile dintre stratul de intrare – stratul ascuns şi stratul ascuns – stratul de ieşire, precum şi numărul parametrilor reţelei	232
Figura 5.9	Structura neurală rezultată prin aplicarea metodei MOBD. Liniile punctate marchează conexiunile / nodurile eliminate	233
Figura 5.10	Corelaţia între valorile de ieşire din nodurile ascunse, după etapa de învăţare	233
Figura 5.11	Covariaţia dintre semnalul de ieşire din fiecare nod ascuns şi semnalul de eroare la ieşirea reţelei	234
Figura 5.12	Vectorii proprii reciproc ortogonali, ai matricii $\boldsymbol{Z}^T \boldsymbol{Z}$, după etapa de învăţare ($\boldsymbol{Z}$ este matricea valorilor de ieşire din nodurile ascunse)	234
Figura 5.13	Corelaţia valorilor *net* înainte de intrarea în stratul ascuns, corelaţia valorilor de ieşire din stratul ascuns, corelaţia valorilor ponderilor conexiunilor ce intră în stratul ascuns şi corelaţia valorilor ponderilor după aplicarea transformării neliniare sigmoidale	235
Figura 5.14	Semnalul de ieşire obţinut după descompunerea SVD a matricii de ponderi dintre stratul ascuns şi cel de ieşire, semnalul de ieşire dorit şi eroarea pătratică medie	236
Figura 5.15	Structura reţelei obţinute după descompunerea SVD a a matricii ponderilor conexiunilor dintre stratul ascuns şi cel de ieşire	236
Figura 5.16	Semnalul de ieşire obţinut de structura 5 – 1 – 1, rezultată prin aplicarea metodei SVD–MOBD, semnalul de ieşire dorit şi eroarea pătratică medie	237
Figura 5.17	Valorile ponderilor şi salienţelor în structura 5 – 1 – 1, obţinută prin aplicarea metodei SVD–MOBD	237
Figura 5.18	Structura neurală 5 – 1 – 1 obţinută prin aplicarea metodei SVD–MOBD	238
Figura 5.19	Semnalul de ieşire obţinut şi semnalul de ieşire dorit, după realizarea descompunerii SVD a matricilor de ponderi	238

Scanned with OKEN Scanner

Figura 5.20	Semnalul de ieşire obţinut, semnalul de ieşire dorit şi eroarea pătratică medie după aplicarea metodei TOBD	239
Figura 5.21	Reţeaua neurală obţinută prin aplicarea metodei TOBD. Liniile punctate marchează conexiunile / nodurile eliminate	239
Figura 5.22	Semnalul de ieşire obţinut, semnalul de ieşire dorit şi eroarea pătratică medie de test, după aplicarea metodei OBD standard	240
Figura 5.23	Semnalul de ieşire obţinut după aplicarea metodei OBD standard şi semnalul de ieşire dorit (eşantioane cu indexul în intervalul 1500 – 1575)	240
Figura 5.24	Curba de reînvăţare în metoda OBD. Vârfurile curbei indică eliminarea conexiunilor	241
Figura 5.25	Reţeaua neurală rezultată prin aplicarea metodei OBD. Liniile punctate marchează conexiunile / nodurile eliminate	241
Figura 5.26	Erorile de învăţare şi de test obţinute pentru metodele OBD şi MOBD	246
Figura 5.27	Comparaţie între ponderile conexiunilor dintre nodurile stratului de intrare şi nodurile stratului ascuns, obţinute prin aplicarea metodelor MOBD şi OBD	246
Figura 5.28	Comparaţie între ponderile conexiunilor dintre nodurile stratului ascuns şi nodul de ieşire, obţinute prin aplicarea metodelor MOBD şi OBD	248

Figura 6.1	Zgomotul nefiltrat (200 eşantioane)	276
Figura 6.2	Zgomotul filtrat (200 eşantioane)	276
Figura 6.3	Potenţialul evocat corect şi zgomotos (200 eşantioane)	276
Figura 6.4	Potenţialul evocat corect şi zgomotos (576 eşantioane)	276
Figura 6.5	Influenţa numărului de patternuri	281
Figura 6.6	PSNR şi numărul epocilor de învăţare, în funcţie de dimensiunea ferestrei W	280
Figura 6.7	Influenţa numărului de noduri ascunse	281
Figura 6.8	Potenţialul evocat obţinut în cazul normei L_2	281
Figura 6.9	Curba de învăţare pentru funcţiile Huber, Hampel	281
Figura 6.10	Potenţialul evocat (norma L_2, exp. 5)	282
Figura 6.11	Potenţialul evocat estimat (funcţia Huber, exp. 5)	282
Figura 6.12	Potenţialul evocat estimat (funcţia logistică, exp. 5)	282
Figura 6.13	Potenţialul evocat estimat (funcţia Hampel, exp. 5)	282
Figura 6.14	PSNR la ieşirea reţelei în etapa de test, pentru funcţiile cost L_2, Huber şi logistică	283
Figura 6.15	Potenţialul evocat corect şi potenţialul evocat cu zgomot	285
Figura 6.16	PSNR la ieşirea reţelei, în etapa de test, pentru funcţiile	

Scanned with OKEN Scanner

	cost L_2, Hampel şi logistică	285
Figura 6.17	Modificarea parametrului $1 - \theta$, pentru diferite valori $c > 0$	287
Figura 6.18	Semnalul de ieşire obţinut în etapa de test, semnalul de ieşire dorit şi eroarea pătratică medie, pentru cazul funcţiei cost estimator robust, în varianta de optimizare multi – criteriu	287
Figura 6.19	Semnalul de ieşire obtinut, semnalul de ieşire dorit şi eroarea pătratică medie, pentru filtrul median cu fereastra de 3 şi 7 eşantioane	289
Figura 6.20	Semnalul de ieşire obţinut, semnalul de ieşire dorit şi eroarea pătratică medie, pentru filtrul median cu fereastra de 5 şi 9 eşantioane	289
Figura 6.21	Semnalul de ieşire obţinut, semnalul de ieşire dorit şi eroarea pătratică medie, pentru filtrul median cu fereastra de 11 eşantioane	290
Figura A3.1	Modelul perceptronului simplu Fukushima	314
Figura A3.2	Modelul perceptronului simplu Lapedes – Farber	315
Figura A3.3	Modelul perceptronului simplu Frasconi – Gori – Soda	317
Figura A3.4	Modelul perceptronului simplu Frasconi – Gori – Soda generalizat	317
Figura A3.5	Modelul perceptronului simplu Poddar – Unnikrishnan	319
Figura A3.6	Modelul perceptronului simplu Dahanayake – Upton	319
Figura A3.7	Modelul perceptronului simplu Hopfield	320
Figura A3.8	Modelul perceptronului simplu Grossberg	321
Figura A4.1	Funcţia treaptă (prag) "Heaviside" unipolară şi bipolară	322
Figura A4.2	Funcţia prag "cu zonă moartă" şi funcţia prag cu histerezis	323
Figura A4.3	Funcţia liniară (rampă) unipolară şi bipolară	324
Figura A4.4	Funcţia de activare rampă "cu zonă moartă"	324
Figura A4.5	Funcţia de activare modul	324
Figura A4.6	Funcţia de activare trunghiulară	325
Figura A4.7	Funcţia de activare Debenham – Garth	325
Figura A4.8	Funcţii de activare de tip tangentă sigmoidală (unipolară şi bipolară) şi derivatele lor	326
Figura A4.9	Funcţia de activare sigmoidă nemonotonă	327
Figura A4.10	Funcţia de activare gaussiană	328
Figura A4.11	Funcţia $\varphi(r)$ "pălărie mexicană"	328

Scanned with OKEN Scanner

Figura A5.1	Curbele de învățare (eroare de învățare ca funcție de numărul de epoci) pentru norma L_2, diferite valori ale parametrului γ și funcția de activare $f_1(s, net)$	334
Figura A5.2	Curbele de învățare pentru norma L_2, diferite valori ale parametrului γ și funcția de activare $f_2(s, net)$	334
Figura A5.3	Curbele de învățare pentru funcția cost Hampel, $\beta = 0.5$ și funcția de activare $f_1(s, net)$	335
Figura A5.4	Curbele de învățare pentru funcția Hampel, $\beta = 0.5$ și funcția de activare $f_2(s, net)$	335
Figura A5.5	Curbele de învățare pentru funcția cost logistică, $\beta = 0.1$ și funcția de activare $f_1(s, net)$	335
Figura A5.6	Curbele de învățare pentru funcția cost logistică, $\beta = 0.1$ și funcția de activare $f_2(s, net)$	335
Figura A5.7	Curbele de învățare pentru norma L_2, $\gamma = -0.0001$ și funcția de activare $f_1(s, net)$	336
Figura A5.8	Curbele de învățare pentru norma L_2, valori negative ale γ și funcția de activare $f_2(s, net)$	336
Figura A5.9	Curbele de învățare pentru strategiile "flip – flop" și standard, norma L_2, diferite valori ale parametrului γ și funcția de activare $f_1(s, net)$	336
Figura A5.10	Detaliu al figurii A5.9, pentru $\gamma = 0.0001$	336
Figura A5.11	Curbele de învățare pentru strategiile "flip – flop" și standard, norma L_2, diferite valori ale parametrului γ și funcția de activare $f_2(s, net)$	337
Figura A5.12	Curbele de învățare pentru strategiile "flip – flop" și standard, funcția cost logistică, $\beta = 0.1$, diferite valori ale parametrului γ și funcția de activare $f_1(s, net)$	337
Figura A5.13	Curbele de învățare pentru strategiile "flip – flop" și standard, funcția cost logistică, $\beta = 0.1$, diferite valori ale parametrului γ și funcția de activare $f_2(s, net)$	337
Figura A5.14	Detaliu al figurii A5.13.	337

Figura A6.1	Diagrama Hinton a ponderilor conexiunilor dintre stratul de intrare și stratul ascuns, pentru RNA care a învățat semnalul "chirp"	341
Figura A6.2	Diagrama Hinton a ponderilor conexiunilor dintre stratul ascuns și stratul de ieșire, pentru RNA care a învățat semnalul "chirp"	342
Figura A6.3	Semnalul obținut la ieșirea nodului ascuns $h = 1$ și	

Scanned with OKEN Scanner

semnalul determinat pentru acelaşi nod, prin efectuarea rotaţiei, pentru RNA care a învăţat semnalul "chirp" 342
Figura A6.4 Semnalul de ieşire obţinut şi semnalul de ieşire determinat prin efectuarea rotaţiei, pentru reţeaua care a învăţat semnalul "chirp" 343
Figura A6.5 Patternurile de intrare desenate ca vectori şi patternurile obţinute la ieşirile stratului ascuns 343
Figura A6.6 Patternurile obţinute la ieşirea fiecărui nod din stratul ascuns, pentru RNA care a învăţat semnalul "chirp" 344

Scanned with OKEN Scanner

LISTA DE TABELE

Tabelul 2.1	Principalele momente în evoluţia modelelor neurale artificiale	16
Tabelul 2.2	Obiectivele şi concluziile proiectului DARPA	20
Tabelul 2.3	Apariţia primelor aplicaţii ale diferitelor modele neurale	21
Tabelul 2.4	Apariţia primelor implementări ale modelelor neurale	22
Tabelul 2.5	Priorităţi teoretice, software şi hardware în domeniul neural, în funcţie de diferitele zone geografice	22
Tabelul 2.6	Exemple de aplicaţii ale reţelelor neurale artificiale în diferite domenii	27
Tabelul 2.7	Reprezentare simbolică vs. numerică şi cunoştinţe structurate vs. nestructurate	29
Tabelul 3.1	Micro – structuri neurale	33
Tabelul 3.2	Clasificarea funcţiilor de activare	39
Tabelul 3.3	Clasificarea mezo – structurilor "feedforward"	45
Tabelul 3.4	Clasificarea modelelor neurale din punct de vedere al învăţării	50
Tabelul 3.5	Clasificarea legilor de învăţare	51
Tabelul 3.6	Distribuţii apriori posibile ale ponderilor	80
Tabelul 4.1	Dimensionarea stratului ascuns în structurile neurale "feedforward"	86
Tabelul 4.2	Alegerea mărimilor $C_{\hat{i}nv}(\mathbf{w})$ şi χ, pentru diferite criterii statistice, când $C_{complexitate}(\mathbf{w}) = \Pi$	92
Tabelul 4.3	$C_{complexitate}(\Pi)$ pentru criteriile TIC, AIC şi MDL	93
Tabelul 4.4	Algoritmul PMDL	104
Tabelul 4.5	Clasificarea metodelor ontogenice crescătoare, după strategia constructivă	109
Tabelul 4.6	Clasificarea metodelor ontogenice crescătoare după funcţia de activare a nodului ascuns adăugat	109
Tabelul 4.7	Clasificarea structurilor neurale rezultate prin metode constructive	109
Tabelul 4.8	CASCOR vs. Quickprop vs. BKP în problema parităţii 8 – dimensionale	121

Scanned with OKEN Scanner

Tabelul 4.9	Generarea structurilor "promiţătoare"	126
Tabelul 4.10	Crearea unei noi structuri	127
Tabelul 4.11	Rezultatele metodei în aproximarea funcţiilor polinomiale	128
Tabelul 4.12	Realizarea reţelei entropice	128
Tabelul 4.13	Algoritmul Wang	136
Tabelul 4.14	Rezultatele comparaţiei algoritmilor Upstart, CASCOR, LPPA, OSA	138
Tabelul 4.15	Metode ontogenice constructive pentru proiectarea reţelelor neurale "feedforward"	139
Tabelul 4.16	Clasificarea metodelor "distructive" în funcţie de strategia de reducere a structurii (varianta I)	144
Tabelul 4.17	Clasificarea metodelor "distructive" în funcţie de strategia de reducere a structurii (varianta II)	144
Tabelul 4.18	Clasificarea propusă a metodelor ontogenice "distructive"	145
Tabelul 4.19	Paşii algoritmului OBD	152
Tabelul 4.20	Estimarea salienţei în metoda OBS	158
Tabelul 4.21	Algoritmul SSM	159
Tabelul 4.22	Algoritmul metodei bazate pe corelaţie	176
Tabelul 4.23	Algoritmul FARM	177
Tabelul 4.24	Metode ontogenice "distructive" pentru proiectarea reţelelor neurale "feedforward"	179
Tabelul 4.25	Clasificarea metodelor ontogenice mixte	183
Tabelul 4.26	Erorile după 1000 de epoci de învăţare, pentru experimentele 1 şi 2	188
Tabelul 4.27	Rezultate experimentale (EPMN şi PSNR) pentru imaginea de test Lenna, 255 x 255 pixeli, cu zgomot gaussian (structuri fixe)	188
Tabelul 4.28	Rezultatele în etapa de învăţare şi de test, pentru imaginea Lenna, 255 x 255 pixeli, în cazurile structurii variabile cu algoritmul CASCOR, structurii fixe cu algoritmul BKP şi filtrului Wiener	190
Tabelul 4.29	Comparaţie între structura complet interconectată şi cea parţial interconectată obţinută, din punct de vedere al numărului parametrilor şi numărului operaţiilor de "înmulţire + adunare"	191
Tabelul 4.30	Algoritmul G – S	194
Tabelul 4.31	Comparaţia rezultatelor algoritmilor G – S şi BKP pentru problema "două spirale"	196
Tabelul 4.32	Algoritmul SOSLA pentru structura cu DNAL	199
Tabelul 4.33	Rezultatele metodei DNAL pentru problema SAU–EXCLUSIV NEGAT şi comparaţie cu rezultatele empirice	200

Scanned with OKEN Scanner

Tabelul 4.34	Comparaţie generală între diferite tipuri de metode de proiectare a structurilor neurale "feedforward"	201
Tabelul 5.1	Algoritmul MOBD	206
Tabelul 5.2	Algoritmul SVD–MOBD	210
Tabelul 5.3	Algoritmul TOBD	217
Tabelul 5.4	Indicatori de evaluare a performanţelor structurii neurale rezultate	221
Tabelul 5.5	Erori de învăţare, erori de test şi PSNR înainte şi după aplicarea algoritmului MOBD	225
Tabelul 5.6	Erori de învăţare, erori de test şi PSNR înainte şi după aplicarea algoritmului SVD–MOBD	227
Tabelul 5.7	Erori de învăţare, erori de test şi PSNR înainte şi după aplicarea algoritmului TOBD	227
Tabelul 5.8	Erori de învăţare, erori de test şi PSNR înainte şi după aplicarea algoritmului OBD	228
Tabelul 5.9	Numărul epocilor de reînvăţare şi pasul de reducere corespunzător, în algoritmii OBD, MOBD şi TOBD	242
Tabelul 5.10	Eroarea de învăţare, eroarea de test şi PSNR obţinute prin aplicarea algoritmilor MOBD, TOBD şi OBD	243
Tabelul 5.11	Comparaţia modelelor rezultate prin aplicarea metodelor MOBD, SVD–MOBD, TOBD şi OBD cu modelele AR(7) şi ARMAX(7), din punct de vedere al valorii erorii de generalizare, estimată cu criteriul FPE (Akaike)	244
Tabelul 5.12	Scăderea PSNR odată cu micşorarea numărului de parametri, pentru metodele MOBD, SVD–MOBD, TOBD şi OBD	244
Tabelul 5.13	Numărul ponderilor, pragurilor şi numărul total al parametrilor ce trebuie stocaţi după aplicarea metodelor SVD–MOBD, TOBD, MOBD şi OBD	245
Tabelul 5.14	Estimatul numărului parametrilor, conform criteriului FPE	247
Tabelul 5.15	Numărul operaţiilor de "înmulţire + adunare" în etapa de test, pentru structurile rezultate prin metodele MOBD, SVD–MOBD, TOBD şi OBD	248
Tabelul 5.16	Duratele etapelor de testare	249
Tabelul 5.17	Spaţiul necesar implementării structurilor neurale rezultate	249
Tabelul 6.1	Funcţii $\Phi(r)$ estimatori robuşti de tip M	266
Tabelul 6.2	Relaţia de calcul pentru $\delta_j(\xi)$, depinzând de funcţia cost	

Scanned with OKEN Scanner

	aleasă	268
Tabelul 6.3	Experimente desfăşurate	278
Tabelul 6.4	Influenţa numărului de patternuri *P* asupra erorii de învăţare	279
Tabelul 6.5	Influenţa dimensiunii ferestrei *W* asupra performanţelor reţelei	278
Tabelul 6.6	Eroarea pătratică medie normată la ieşire, pentru diferite funcţii cost	283
Tabelul 6.7	Influenţa pantelor adaptive asupra estimării potenţialelor evocate	284
Tabelul 6.8	EPMN în etapa de învăţare (exp. 3) şi EPMN la ieşirea filtrului median, în funcţie de dimensiunea ferestrei de intrare	288
Tabelul 6.9	Numărul operaţiilor "înmulţire + adunare" în etapa de test, pentru RNA şi numărul operaţiilor de comparare, pentru filtrul median	290
Tabelul 6.10	Duratele etapelor de testare pentru RNA şi filtrul median	291
Tabelul 7.1	Criterii de acceptare a RNA ca instrument de PDS	295
Tabelul A1.1	Proiecte în domeniul reţelelor neurale, în mediul industrial (1990 – 1996)	298
Tabelul A1.2	Proiecte în domeniul reţelelor neurale, în mediul universitar (1990 – 1996)	301
Tabelul A2.1	Implementări software ale reţelelor neurale, în mediul industrial şi universitar (1990 – 1996)	307
Tabelul A2.2	Implementări analogice ale reţelelor neurale, în mediul industrial şi universitar (1990 – 1996)	308
Tabelul A2.3	Implementări digitale ale reţelelor neurale, în mediul industrial şi universitar (1990 – 1996)	309
Tabelul A2.4	Implementări optice ale reţelelor neurale, în mediul industrial şi universitar (1990 – 1996)	311
Tabelul A2.5	Implementări hibride ale reţelelor neurale, în mediul industrial şi universitar (1990 – 1996)	312

Scanned with OKEN Scanner

Tabelul A5.1 Influența termenului γ de învățare a pantei, asupra erorii de test în sens L_2 333
Tabelul A5.2 Eroarea de test în sens L_2, pentru $\gamma < 0$ și $\gamma > 0$ 338
Tabelul A5.3 Strategia de învățare "flip – flop" 338

Scanned with OKEN Scanner

1 INTRODUCERE

"Istoria ştiinţei pare să fie martora unei continue pierderi de către om a poziţiei dominante faţă de natură. Odată cu ideile lui Copernic în domeniul astronomiei, omenirea şi-a pierdut locul central în univers. La apariţia teoriei biologice darwiniste, evoluţia oamenilor a devenit similară celei a animalelor. Din punct de vedere al psihanalizei lui Freud, omul nu mai este o fiinţă superioară dominată de raţiune, ci condusă de instincte animalice. Odată cu teoria 'calculatorului cerebral', se pare că omul îşi pierde ultima poziţie de superioritate faţă de natură. Inteligenţa sa este redusă la un mecanism cerebral complex ce poate fi depăşit de "calculatoare inteligente". Astfel, asistăm la o criză mondială a acceptării tehnologice, cauzată uneori de sentimente iraţionale şi teamă. Reducerea ipotezelor ştiinţifice nu înseamnă reducţionism etic. Valoarea omului nu depinde de poziţia sa în evoluţia naturală, de explicarea funcţiilor sale cerebrale sau de simularea acestora pe maşini, ci de valori etice aşa cum apar ele în declaraţia drepturilor omului. Protecţia acestor drepturi este subiectul principal al eticii tehnologiei moderne. Pentru a realiza acest ţel etic, este necesară înţelegerea din ce în ce mai bine, a modului uman de gândire, simţire şi acţionare. In acest sens, epistemologia etică oferă noi perspective şi depăşeşte vechile doctrine. Este o parte a unui program de cercetare interdisciplinar, care împinge mai departe o tehnologie revoluţionară din domeniul calculatoarelor" (Ronny Meir).

1.1 APARIŢIA MODELELOR CONEXIONISTE

1.1.1 Prelucrare paralelă şi distribuită vs. prelucrare secvenţială

Progresul ştiinţific a fost accelerat semnificativ de perfecţionarea calculatoarelor. Evoluţia lor rapidă impune redefinirea conceptului de "maşină", considerau Ritter, Martinez şi Shulten [R4]. Mult timp identificat cu structura secvenţială von Neumann, calculatorul tradiţional nu a reuşit să facă faţă tuturor problemelor care necesită calcul intensiv, precum preprocesarea semnalelor, clasificarea formelor şi recunoaşterea acestora, coordonarea mişcărilor în mecanismele complexe, luarea deciziilor pe baza unor seturi mari de date cu zgomot, grafica de animaţie de înaltă definiţie, etc. A devenit din ce în ce mai evidentă necesitatea utilizării unor alte metode de prelucrare, implementate pe

Scanned with OKEN Scanner

sisteme mai performante. Realizările din domeniul tehnologiei semiconductoarelor, cu perspectiva reducerii costului hardware-ului, au făcut posibilă, în anii '80, aplicarea principiilor de prelucrare paralelă și distribuită a semnalelor [R4, S11]. Inspirate din funcționarea modelelor biologice, sistemele bazate pe acestea au avut un impact puternic pe de o parte asupra domeniului tehnic, prin crearea unor noi instrumente și, pe de altă parte, "au oferit speranța realizării unei simulări eficiente și suficient de precise a fenomenelor cunoașterii umane" [R3].

Prelucrarea paralela si distribuita reprezinta o noua perspectiva asupra perceptiei, memoriei, învatarii si gândirii — si asupra mecanismelor de calcul ce stau la baza prelucrarii inteligente a informatiei în general (Rumelhart si McClelland).

Ideea prelucrării paralele distribuite (PDP) — că inteligența ia naștere din interacțiunea dintre un număr mare de elemente de prelucrare simple — apăruse cu mult timp în urmă și a contrastat cu convingerile existente legate de caracteristicile percepției umane, memoriei, limbajului și gândirii. Instrumentele formale acceptate pentru caracterizarea proceselor mentale au devenit vizibile [H508, P201]. Reflectate în sistemele de calcul realizate, ideile PDP au sugerat utilizarea unui număr mare de elemente de prelucrare simple interconectate.

Prezentarea ideilor preluate din domeniul biologic, ce au stat la baza sistemelor PDP impune menționarea caracteristicilor sistemului nervos biologic și a concluziilor desprinse din paralela între acesta și sistemele de calcul existente.

1.1.2 Inspirația din neuroștiințe

Neuroștiințele reprezintă un ansamblu de discipline științifice al căror obiect de studiu îl reprezintă înțelegerea creierului uman [H7].

1.1.2.1 Sistemul nervos biologic

Structura sistemului nervos biologic reprezintă obiectul de studiu al biologiei, iar funcționarea sa al neurobiofizicii. Neurofiziologia, pe de altă parte, cercetează modul de trecere de la procesele la nivel neural la funcțiile integrative ale sistemului nervos [V200, W10]. Rezultatele tuturor acestor științe au fost utilizate pentru construirea modelelor conexioniste.

Neuronul sau celula nervoasă este unitatea structurală și funcțională a sistemului nervos. Numărul neuronilor este estimat la circa 10^{10}, fiecare fiind conectat cu alți 10^4 neuroni [B9]. Dimensiunile și formele acestora sunt foarte

Scanned with OKEN Scanner

diferite. In ciuda diversităţii lor, la nivel structural majoritatea neuronilor se compun din corpul sau soma neurală (pericarion), prelungiri dendritice şi axon [H11].

Funcţionarea schematică a neuronului biologic constă din sumarea algebrică în arborele dendritic, a semnalelor (potenţiale electrice pre–sinaptice) primite de la neuronii înconjurători şi transmiterea rezultatului către corpul celulei. Dacă valoarea sa depăşeşte un prag (10, până la 20 mV), atunci corpul celulei produce un impuls electric (numit post – sinaptic sau potenţial de acţiune), transmis în lungul axonului. Potenţialul de acţiune este codificat în amplitudine la nivelul membranei post–sinaptice şi în frecvenţă pe canalul de transmisie.

Sistemul nervos central are la bază creierul uman [R4, B2], compus din două structuri multistrat, numite emisfere cerebrale. Stratul exterior (cortex cerebral) este deosebit de complex şi, din punct de vedere evolutiv, este partea cea mai recent dezvoltată a creierului. Cortexul uman constă dintr-un strat de celule nervoase cu aria de aproximativ 0.2 m^2 şi grosime de aproximativ 2 – 3 milimetri, convolut puternic. Printre caracteristicile sale se pot enumera: ❶ localizarea structurală şi funcţională; specializarea; ❷ paralelismul structural şi funcţional; ❸ auto – organizarea; ❹ prelucrarea secvenţială şi prelucrarea distribuită; ❺ performanţele epistemice.

Codarea şi prelucrarea informaţiei în sistemul nervos biologic se bazează pe principii care nu sunt pe deplin elucidate. Nivelul de bază al codării se află la nivelul codului genetic al fiecărei celule. Aceasta conţine informaţia genetică în secvenţele de aminoacizi din moleculele proteinice. Experimente realizate între anii 1944 – 1953, au identificat în acidul ADN, materialul genetic de bază. Codul este compus din patru elemente de cod, iar o secvenţă adiacentă de trei elemente de cod constituie un cuvânt de cod [S11, S43].

Al doilea nivel de codare în sistemele biologice se află la nivelul citoplasmei fiecărei celule, în polimeri de formă cilindrică. Se pare că acest cod este răspunzător, cel puţin în parte, de comunicarea la nivel biologic, prelucrarea informaţiei, procesele cognitive şi de gândire.

Intre acidul ADN şi neuron se află alte nivele de codare, după care urmează codarea realizată de cel din urmă. La nivelul creierului uman, codarea informaţiei poate fi studiată pe baza teoriilor şi modelelor elaborate pentru acesta.

Proprietăţile esenţiale ale prelucrării umane a informaţiei sunt: ❶ prelucrare paralelă şi distribuită; ❷ prelucrarea informaţiei temporale codificate (trenuri de impulsuri); ❸ prezenţa mecanismelor de activare / inhibiţie; ❹ degradarea lentă a informaţiei şi utilizarea redundanţei; ❺ stocarea informaţiilor în mod distribuit în memorie adresabilă prin conţinut; ❻ proces iterativ de recuperare a informaţiei; ❼ ieşire continuă.

Scanned with OKEN Scanner

1.1.2.2 Comparaţie între creierul uman şi sistemul de calcul

In anul 1958, J. von Neumann sublinia că, limbajul matematic al calculatoarelor digitale controlate prin program era bine stabilit, în timp ce modelarea matematică a creierului nu s-a realizat încă. Deşi fusese vizualizat prin metode tehnice avansate, încă nu exista o teorie a "calculatorului cerebral". Era necesară mai întâi explicarea fenomenelor de auto – organizare, de evoluţie a sistemelor complexe, etc. Din punctul de vedere al filozofiei ştiinţei, programul de cercetare în acest domeniu este reducţionist, bazat pe ipoteze simplificatoare.

Pentru îmbunătăţirea modelelor existente ale sistemului nervos central, este utilă simularea sa cu ajutorul structurilor artificiale. Pornind de la observaţia că, există multe trăsături ale creierului care sunt utile sistemului artificial (❶ este robust şi tolerant la defecte; ❷ este flexibil şi se poate adapta la un nou mediu prin "învăţare"; ❸ poate opera cu informaţii care sunt fuzzy, probabilistice, cu zgomot sau inconsistente; ❹ este o structură puternic paralelă; ❺ are dimensiuni mici şi este compact), se poate realiza comparaţia sa schematică cu sistemele de calcul secvenţiale, urmată de includerea acestor caracteristici în modelele artificiale. Perfecţionarea lor se realizează prin preluarea unor trăsături din ce în ce mai complexe ale sistemului biologic de referinţă, iar rezultatele se reflectă în final, în înţelegerea sa mai profundă.

1.1.2.3 Ideile preluate din neuroştiinţe

Sistemul neural biologic a sugerat o serie de idei, printre care: paralelismul, distribuirea informaţiei, structura multistrat, învăţarea de tip auto – organizare, adaptabilitatea, toleranţa la defecte [H3, S11].

Prelucrarea paralelă şi distribuită ca alternativă a utilizării modelului secvenţial von Neumann, se bazează pe o structură de prelucrare (model alternativ), constituită dintr-un mare număr de elemente simple de prelucrare, care preiau următoarele caracteristici ale neuronului biologic [B9]: ❶ funcţia de bază a neuronului biologic, de sumare a semnalelor de intrare şi furnizare a unui semnal de ieşire, în funcţie de rezultatul comparării sumei cu un nivel de prag; ❷ semnalul de ieşire depinde de semnalele de intrare; ❸ eficienţa sinapselor este modelată prin introducerea unui factor multiplicativ, numit pondere, pentru fiecare dintre intrările neuronului.

1.2 DOMENIUL NEUROINFORMATICII

Noua abordare a prelucrării semnalelor, care nu se bazează pe formularea algoritmică a setului de reguli, se numeşte neuroinformatică [H7, R3, L4]. Neuroinformatica este o componentă a "ştiinţelor complexităţii" ("sciences of

Scanned with OKEN Scanner

complexity") care "vor ocupa centrul vieţii intelectuale a secolului 21" şi reprezintă o sinteză a ştiinţelor tradiţionale precum biologia, fizica, matematica, ştiinţa calculatoarelor, un rezultat al noilor idei, întrebări şi tehnologii recent dezvoltate [H19].

Modelele artificiale ce stau la baza neuroinformaticii sunt modele de prelucrare paralelă şi distribuită, numite şi modele conexioniste, modele neurale artificiale sau modele neuromorfice. Teoria generală creată se numeşte conexionism. Cercetarea în domeniul neuroinformaticii include aspecte legate de teorie şi arhitecturi[1], implementări şi aplicaţii.

1.2.1 Definiţiile modelelor neurale artificiale

Modelele neurale artificiale sau reţelele neurale artificiale (RNA) au fost definite în multe moduri. Cercetătorului Wesley R. Elsberry [E200] i se datorează unul dintre răspunsurile la întrebarea "ce sunt RNA ?":

> *Puritanii ar spune ca o retea neurala este o colectie de tesuturi ectodermice raspunzatoare de activitatea nervoasa a animalului. Altii ar afirma ca, retelele neurale artificiale sunt o metoda de calcul, ale carei proprietati depind de utilizarea unor procesoare de baza, puternic interconectate. De obicei, conexiunile dintre acestea pot fi modificate. De obicei, retelele neurale artificiale realizeaza o prelucrare neliniara. Si tot de obicei, este dificil sa se faca orice afirmatie despre retelele neurale artificiale, deoarece exista atât de multe modele si atât de multe presupuneri si metode de lucru.*

Michael Arbib [E200] a făcut o afirmaţie contradictorie, caracterizată drept "un joc de cuvinte pur şi simplu înspăimântător", care exprimă însă, lipsa de prejudecăţi ce trebuie să caracterizeze cercetătorul în domeniul reţelelor neurale:

> *Creierul uman nu este un calculator pe baza de cereale numai fiindca arata ca o budinca.*

Definiţia 1.1 RNA sunt un instrument de analiză, inspirat din structura paralelă a creierului uman. Simulează un model de prelucrare paralel, puternic interconectat, compus din multe elemente de procesare relativ simple [E5].

Definiţia 1.2 Sistemele de dimensiuni mari, ce pot învăţa adaptiv, se numesc sisteme neurale artificiale [S11].

[1] In neuroinformatică, cuvântul "arhitectură" este utilizat pentru a indica definiţia matematică formală a structurii artificiale [H7].

Scanned with OKEN Scanner

Definiția 1.3 RNA sunt sisteme distribuite software sau hardware de prelucrare a semnalului, alcătuite dintr-un număr mare de elemente de procesare simple (EP), numite neuroni artificiali (perceptroni, neurode, etc), conectate în conformitate cu o anumită topologie și având capacitatea de a-și modifica din punct de vedere cantitativ conexiunile, precum și parametrii proprii de prelucrare [C16].

Definiția 1.4 RNA este o structură paralelă și distribuită de prelucrare a semnalului, compusă din elemente de prelucrare (care pot avea memorie locală și pot prelucra informație localizată), interconectate prin intermediul unor canale de semnal, numite conexiuni. Fiecare element de prelucrare are o singură conexiune de ieșire, semnalul respectiv putând avea orice formă matematică. Prelucrarea ce se desfășoară la nivelul fiecărui element poate fi definită arbitrar, cu restricția să fie în totalitate locală, adică să depindă numai de semnalele de intrare ale elementului de prelucrare și de valorile stocate în memoria sa locală [H7].

Definiția 1.5 RNA reprezintă o clasă de algoritmi [Z2].

Definiția 1.6 RNA sunt o reprezentare (notație) grafică pentru o clasă mare de algoritmi [P3].

Domeniul RNA se desfășoară între două extreme, de aceea și definițiile 1.1 – 1.6 sunt diferite, reprezentând alternative posibile. La o extremă se află cercetători care încearcă să realizeze modele biofizice detaliate și realiste ale neuronilor, scopul final fiind simularea funcțiilor superioare și crearea unor modele ale performanțelor umane. La cea de-a doua extremă se plasează cercetători care propun utilizarea RNA ca instrumente de prelucrare a semnalului (modele constituite din elemente de prelucrare simple, interconectate), ca modele matematice abstracte sau clase de algoritmi, aplicabile în rezolvarea problemelor dificile de abordat prin metode tradiționale.

1.2.2 Terminologie

Sursele de inspirație diferite (biologice, din fiziologie și psihologie) și dezvoltarea independentă a domeniului în comunități științifice precum cea neurobiologică, matematică, fizică, știința calculatoarelor, etc. au avut drept efect o terminologie diversificată și o nomenclatură sinonimică (exemplu: modele conexioniste, de prelucrare paralelă distribuită, neuromorfice, etc. sunt denumiri echivalente), în paralel cu apariția definițiilor independente. Stabilirea unei terminologii unice este esențială, pentru a permite dezvoltarea "verticală" față de cea "orizontală" (abordări noi, posibil diferite) a domeniului. Eforturile

Scanned with OKEN Scanner

comitetului de standardizare în domeniul rețelelor neurale, ca parte a IEEE Neural Network Society, este una dintre puținele tentative semnificative de standardizare, căreia i se adaugă cea realizată de Fiesler [F37]. Un pas important spre standardizare este utilizarea unor notații consistente și convenabile, caracterizate prin simetrie[2] și mnemonicitate[3].

Denumirea pe care o vom utiliza în continuare este cea de "rețele neurale artificiale" (RNA). Varianta alternativă "rețele neurale" se referă de asemenea, la modele artificiale, cu excepția cazurilor în care există alte precizări. De asemenea, vom evita utilizarea termenilor din domeniul biologic: neuron, sinapsă, etc. în favoarea celor de perceptron simplu, nod, neurodă, conexiune, etc. [H19].

1.2.3 Caracteristicile și limitele modelelor neurale artificiale

Caracteristicile modelelor conexioniste (rețele neurale artificiale) sunt: paralelism și control local, reprezentare distribuită, redundanță și adaptabilitate. Prezentarea RNA se poate realiza de la modelul cel mai simplu spre cel complex, sau prin simplificarea treptată a unui model general propus inițial.

Interesul inițial pentru domeniul rețelelor neurale artificiale s-a datorat în mare măsură afirmației că, acestea pot explica mecanismele de bază și dinamica creierului. RNA actuale sunt foarte departe de structura neurală biologică și de simularea proceselor cognitive. Diferențele existente se referă atât la arhitectură, cât și la proprietăți și pot fi caracterizate ca limite tehnice și limite etice. Din prima categorie fac parte următoarele:

❶ Rețelele neurale artificiale nu copiază structurile neurale biologice, ci reprezintă modele mult simplificate ale acestora. "Intrucât nici cele mai realiste modele ale creierului s-ar putea să nu reușească evidențierea funcționării țesutului nervos, studiul trebuie realizat pe modele simplificate, care să conțină trăsături esențiale" (T. Sejnowski). Rumelhart și McClelland completează: "credem că, adevărata cheie a progresului științific este realizarea aproximațiilor și a simplificărilor corecte". Pe de altă parte, deși "rețelele neurale sunt fundamental structuri simple" [H7], modelul nu poate fi simplificat oricât de mult.

❷ Intre modelele neurale artificiale și cele biologice există o serie de deosebiri [D5, E5]:

■ RNA sunt realizate de obicei pe baza câtorva tipuri de neuroni artificiali, în timp ce există o mare varietate de neuroni biologici.

■ Există o diferență semnificativă între numărul neuronilor artificiali din

[2] Simetria presupune ca entități similare să aibă notații similare.
[3] Mnemonicitatea presupune ca simbolul sau variabila aleasă să aibă un nume care să fie ușor memorat (de aceea, se alege astfel încât să-i reflecte, pe cât posibil, semnificația).

Scanned with OKEN Scanner

RNA existente și numărul neuronilor biologici.

- RNA au de obicei topologii de interconectare regulate, în timp ce în creierul uman multe conexiuni par aleatoare sau statistice. Este posibil să existe o precizie mare la nivel celular și al ansamblurilor de celule, ca și la nivel de sistem.
- Există câteva tipuri de conexiuni în RNA și o mare varietate de sinapse biologice.
- Neuronii artificiali sunt mai rapizi decât cei biologici. Primii operează în timpi de ordinul nanosecundelor sau mai puțin, cei din urmă în timpi de ordinul milisecundelor.
- Majoritatea RNA sunt sincrone, în timp ce sistemele biologice realizează modificările asincron.
- Funcționarea RNA se supune legii "celor trei R": aRitmetic, scRiere și citiRe.

1.2.4 Abordări ale modelelor neurale artificiale și probleme de studiu

Temele abordate de lucrările în domeniul rețelelor neurale artificiale, permit clasificarea acestora în două categorii: lucrări ce tratează ❶ aspectele teoretice, scopul lor fiind de prezentare a proprietăților modelelor neurale artificiale existente, de investigare a aspectelor legate de arhitecturi, legi de învățare, etc. și ❷ lucrări de aplicații ale RNA, care se concentrează asupra prezentării utilizării modelelor și asupra soluțiilor pe care le oferă problemelor. Indiferent de tipul teoretic sau aplicativ al abordării, este esențială precizarea obiectului de studiu, în cazul de față rețea neurală artificială, ca instrument de prelucrare a semnalelor. Principalele capitole de studiu în domeniul rețelelor neurale artificiale sunt prezentate în fig. 1.1.

Caracterul interdisciplinar al domeniului RNA se reflectă și în metodele și instrumentele utilizate pentru descrierea formală. Acestea sunt bazele matematice ale prelucrării semnalelor, statistica matematică, termodinamică și teoria învățării.

1.2.5 Modelele neurale artificiale ca instrumente de prelucrare digitală a semnalelor

Obiectul de studiu al prelucrării digitale a semnalelor (PDS) este reprezentarea digitală a acestora, utilizarea procesoarelor digitale pentru analiza, modificarea și extragerea informației din semnale. Printre avantajele PDS se numără: acuratețea prelucrării și flexibilitatea sistemelor, în comparație cu cele analogice, iar printre dezavantaje: timpul de proiectare al sistemelor digitale și problemele legate de lungimea finită a cuvintelor de reprezentare [L200, H21].

Scanned with OKEN Scanner

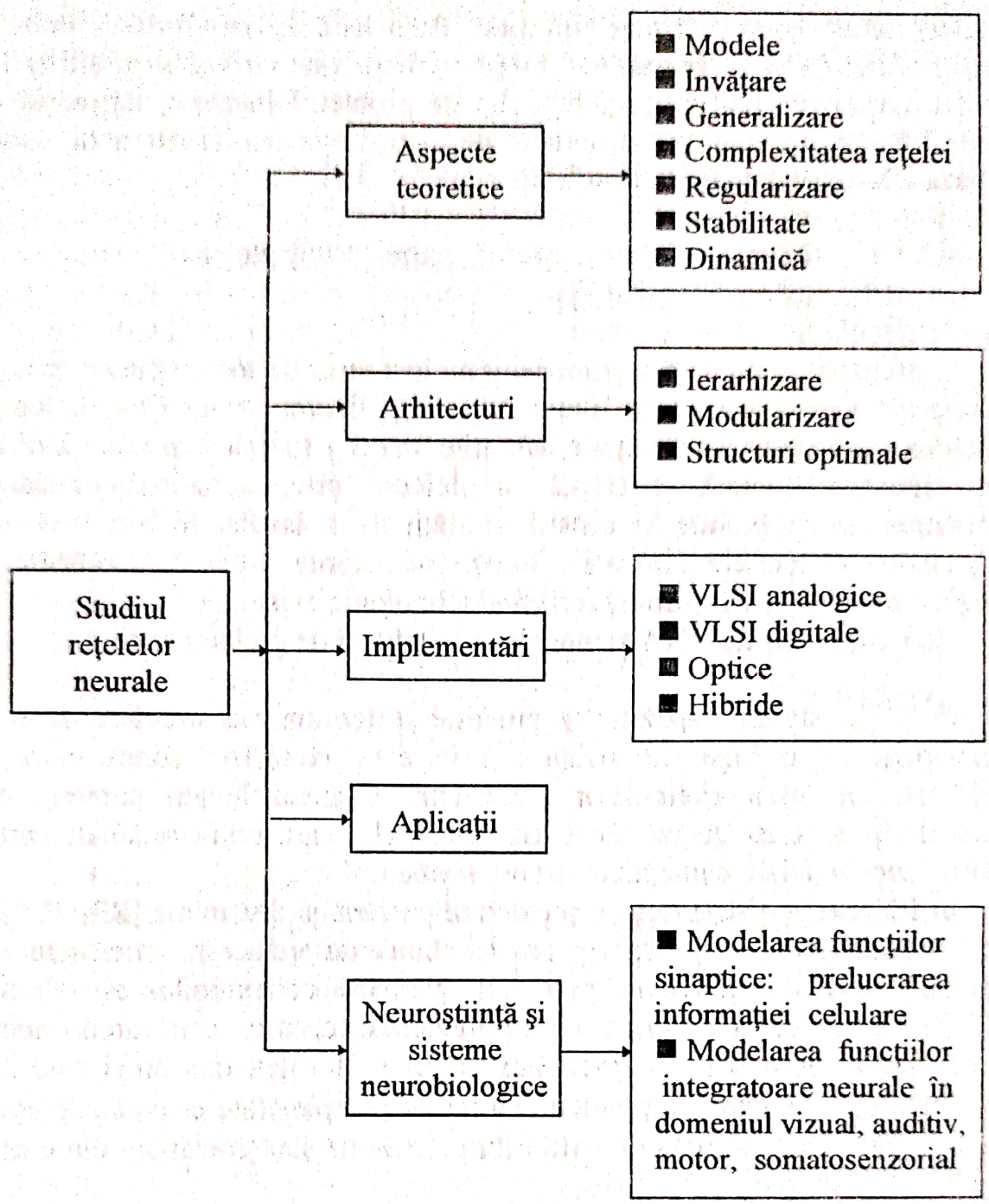


Fig. 1.1 Principalele capitole de studiu în domeniul reţelelor neurale

De asemenea, sistemele clasice de PDS nu pot realiza prelucrare prin învăţare pe baza unor exemple, nu pot generaliza, robusteţea lor este redusă. O alternativă de prelucrare digitală a semnalelor care depăşeşte aceste limite este cea bazată pe RNA [D205]. Dacă interpretarea modelelor biologice pe baza celor artificiale existente este încă problematică ("Incă am îndoieli. Nu ştiu cum să proiectez înţelegerea mea din domeniul cunoaşterii în aceste noi structuri", afirmau Rumelhart şi McClelland), ca instrument de prelucrare, "reţelele neurale

Scanned with OKEN Scanner

sunt cel de-al doilea 'cel mai bun mod' de a face aproape orice", după cum remarca John Denker. Primul 'cel mai bun mod' este de a găsi şi utiliza legile corecte sau algoritmul optim pentru fiecare problemă în parte, dar acest lucru poate fi foarte costisitor şi consumator de timp. De aceea, prelucrarea numerică pe baza RNA este pe deplin justificată.

1.2.5.1 Caracteristicile modelelor neurale artificiale ca instrumente de PDS

In prelucrarea digitală a semnalelor au fost utilizate toate tipurile de modele neurale artificiale. Acestea se obţin prin simplificarea structurilor biologice şi preluarea trăsăturilor lor esenţiale, amintite anterior (paralelism şi control local, reprezentare distribuită, toleranţă la defecte prin codarea redundantă a informaţiei, adaptabilitate în sensul abilităţii de a învăţa pe baza datelor de învăţare sau a experienţei iniţiale). In plus, structurile obţinute operează în timp real şi se pot realiza cu uşurinţă utilizând tehnologia existentă.

Deosebirile dintre RNA şi alte clase de tehnici de prelucrare sunt:

❶ RNA este un dispozitiv de procesare (algoritm sau structură hardware), caracterizat de o lege de învăţare, prin care ponderile conexiunilor sunt modificate pe baza exemplelor prezentate. Reţeaua învaţă numeric şi nu simbolic, în contrast cu metodele convenţionale, care obţin rezultatul printr-o aplicare secvenţială a comenzilor asupra simbolurilor.

❷ RNA este o structură cu prelucrare paralelă şi distribuită [R3, P27], care diferă semnificativ de arhitecturile paralele tipice de prelucrare, prin faptul că, în structură există interconectare puternică. Numărul conexiunilor este de obicei mai mare decât cel al elementelor de prelucrare, ceea ce este rareori adevărat pentru arhitecturile paralele existente. In plus, acestea din urmă tind să fie constituite din elemente de prelucrare a căror complexitate este comparabilă cu cea a maşinilor von Neumann. Arhitecturile neurale sunt constituie din elemente de prelucrare simple.

❸ RNA este o structură tolerantă la defecte.

❹ Majoritatea algoritmilor de învăţare a RNA se compun din câteva operaţii simple de tip adunare + înmulţire, în timp ce modelele tradiţionale von Neumann aveau un singur element central de procesare (CPU), care realiza un număr mare de operaţii, în mod secvenţial.

1.2.5.2 Etapele proiectării modelelor neurale artificiale pentru PDS

"Proiectarea reţelelor neurale artificiale pentru a rezolva probleme [...] poate schimba modul nostru de a gândi şi ne poate conduce spre noi perspective şi algoritmi mai buni", considera Richard Lippmann. Pe de altă parte,

Scanned with OKEN Scanner

Studiul nici unui sistem fizic nu este complet fara descrierea matematica, sau, cu cuvintele lui Galileo Galilei (1564 – 1642): "Cartea naturii este scrisa în limbajul matematicii" (Nabet si Pinter).

Proiectarea RNA se referă la determinarea reţelelor cu dimensiuni reduse, capabile să rezolve o problemă dată [R3]. Caracteristicile pe care trebuie să le aibă un sistem bazat pe RNA sunt: să fie eficient, performant, extensibil, să poată fi inclus ca subsistem în structuri hibride de mari dimensiuni, să fie uşor de utilizat, preferabil "user friendly". Dezvoltarea unui sistem de prelucrare neural este prezentată în fig. 1.2.

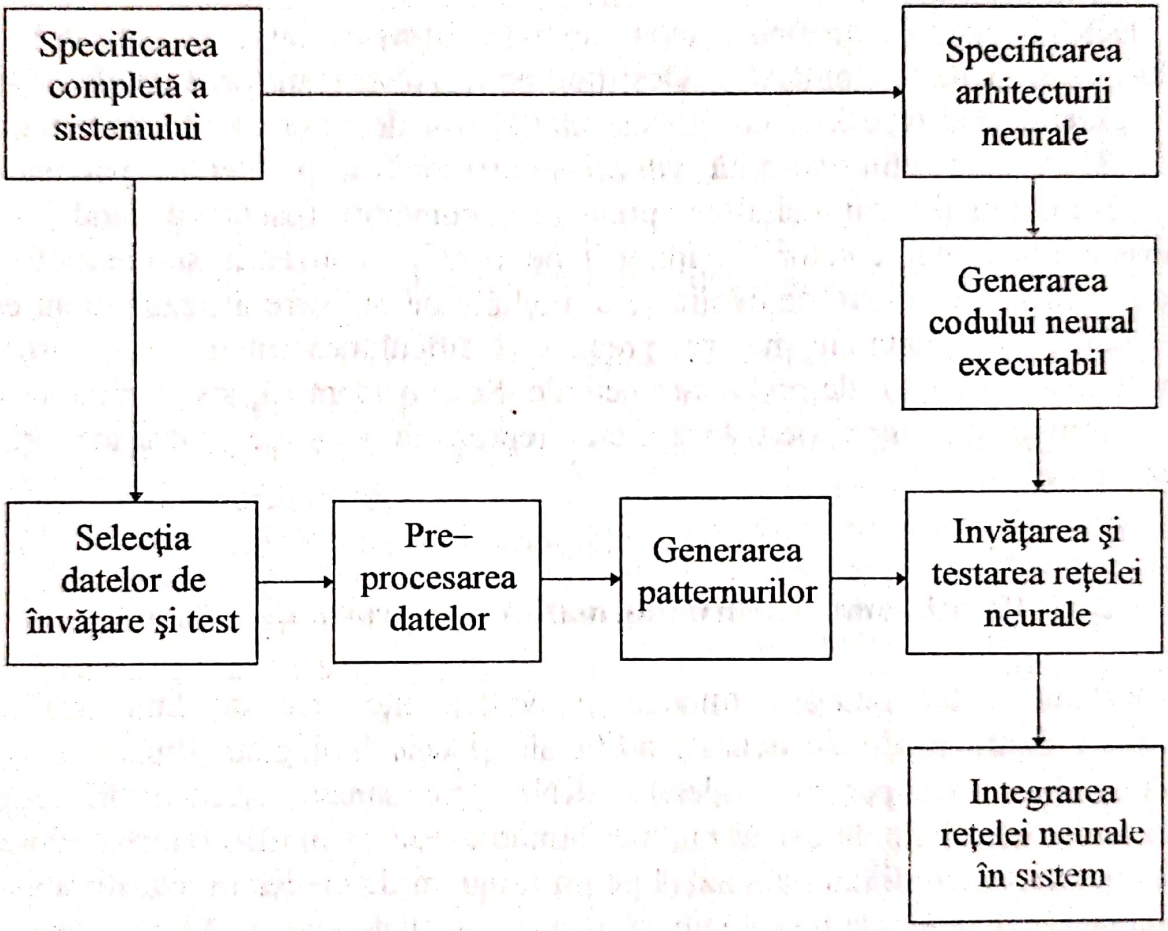


Fig. 1.2 Dezvoltarea unui sistem de prelucrare neural

1.2.5.3 Implementări ale modelelor neurale artificiale

Soluţionarea problemelor reale cu RNA se bazează pe prelucrarea unor volume mari de date, iar scopul este obţinerea unui instrument de lucru care să funcţioneze în timp real. De aceea, o caracteristică a modelelor neurale artificiale este puterea ridicată de calcul pe care o necesită implementarea lor.

Scanned with OKEN Scanner

Alegerea structurilor hardware pentru implementarea RNA presupune utilizarea paralelismului structurilor şi algoritmilor acestora.

Din punct de vedere al implementărilor software, RNA pot fi descrise la trei nivele de limbaj: limbaj de specificaţii de nivel înalt, limbaj de prelucrare paralelă de nivel intermediar (Ada, ANSPEC, OCCAM, C++) şi limbaj de programare convenţional de nivel scăzut (C, Pascal, asamblare). Din punct de vedere al operaţiilor de calcul executate, algoritmii de învăţare în RNA se compun din operaţii simple (deseori adunare şi înmulţire), mai puţine ca număr decât în cazul algoritmilor PDS implementaţi cu sisteme standard.

1.2.5.4 Utilitatea modelelor neurale artificiale ca instrumente de PDS

Reţelele neurale artificiale au obţinut performanţe bune în aplicaţiile de prelucrare digitală a semnalelor. Deşi nu "pot rezolva toate problemele dificile din inginerie, mai repede şi cu un cost mai scăzut decât orice altceva anterior" [E5], RNA sunt utile datorită vitezei (consecinţă a prelucrării paralele şi distribuite), funcţionării neliniare, prelucrării numerice (esenţială când nu se cunoaşte distribuţia datelor de intrare) pe care o realizează şi toleranţei la defecte. Lipsa capacităţii de explicare a modului de obţinere al rezultatului este unul dintre dezavantajele majore, precum şi dificultatea integrării în sisteme complexe a modulelor de prelucrare neurale. Se consideră că, sistemele hibride, care conţin şi structuri expert, fuzzy, etc., reprezintă o soluţie pentru rezolvarea acestei probleme.

1.2.6 Relaţia dintre neuroinformatică şi neuroştiinţe

Relaţia dintre neuroinformatică şi neuroştiinţe este de fapt, legătura complexă dintre modelele neurale artificiale şi cele biologice. Preluarea unor concepte biologice pentru modelarea RNA, menţionată anterior, în scopul obţinerii unor sisteme de calcul mai performante este o abordare epistemologică din punct de vedere filozofic, bazată pe un program de cercetare interdisciplinar (matematică, fizică, neurobiologie şi ştiinţa calculatoarelor). Multe concepte precum paralelismul, neliniaritatea, auto – organizarea, idei din teoria sistemelor complexe, "fuzionează în noul model de conexionism, care la rândul său oferă noi explicaţii naturii şi minţii umane" [M73].

Există însă şi afirmaţii care consideră că nu există nici o relaţie directă între RNA şi modelele biologice. Acestea argumentează prin limitele reţelelor neurale artificiale existente. Ideea că studiul RNA este condiţionat de cunoaşterea structurii şi funcţionării sistemului nervos poate fi, de asemenea, combătută, prin considerarea modelelor neurale artificiale drept instrumente alternative de prelucrare a semnalelor. Este adevărat însă, că descrierea arhitecturilor neurale

Scanned with OKEN Scanner

artificiale nu este completă fără menţionarea caracteristicilor sistemului biologic ca sursă de inspiraţie [D5].

Preluarea ideilor din neuroştiinţă de către neuroinformatică reprezintă numai un aspect al problemei, pentru că există şi o conexiune inversă. Noile arhitecturi, concepte şi teorii dezvoltate pentru modelele artificiale sunt utilizate de cercetătorii din domeniul neuroştiinţei ca noi paradigme pentru creierul uman.

1.3 STRUCTURA LUCRĂRII

Structura lucrării este prezentată în fig. 1.3.

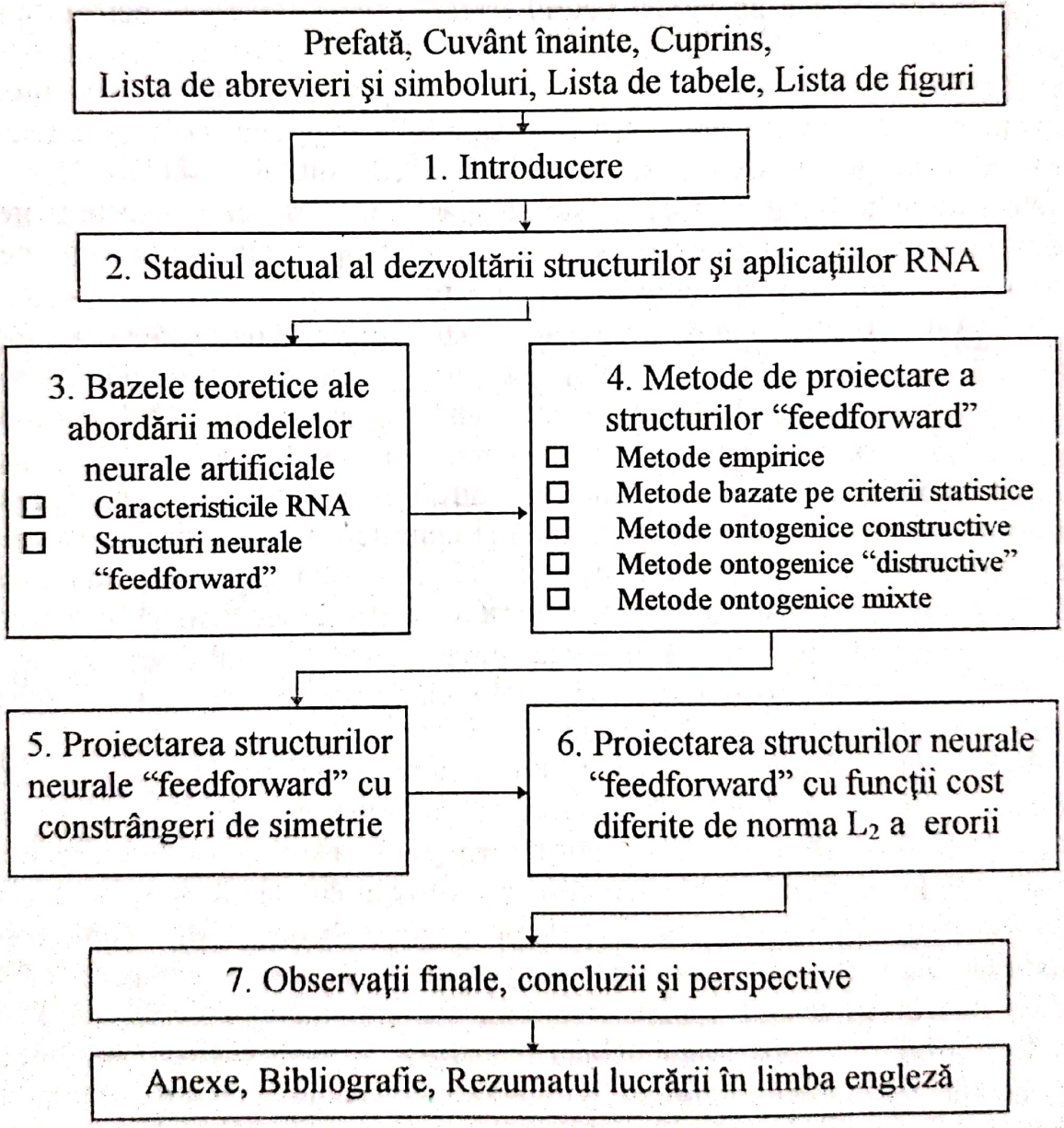


Fig. 1.3 Structura lucrării

Scanned with OKEN Scanner

Capitolul 1 ("Introducere") se referă la aspecte legate de apariţia conexionismului, definirea modelelor conexioniste (reţele neurale artificiale), caracteristicile şi limitele acestora, precum şi aspecte legate de utilizarea RNA ca instrumente de prelucrare digitală a semnalelor. Plasarea lor în contextul neuroinformaticii şi inteligenţei artificiale, precum şi relaţia cu neuroştiinţele sunt de asemenea amintite.

Capitolul 2 ("Stadiul actual al dezvoltării structurilor şi aplicaţiilor reţelelor neurale artificiale"), **anexele 1** şi **2** conţin o prezentare succintă a perspectivei trecut – prezent – viitor în domeniul RNA, din punct de vedere al structurilor, aplicaţiilor şi implementărilor acestora.

Capitolul 3 ("Baze teoretice ale abordării modelelor neurale artificiale") cuprinde două părţi: "Caracteristicile modelelor neurale artificiale", se referă la elementele ce definesc un model neural artificial (caracteristicile perceptronului simplu, topologiile RNA, criterii de cost şi modelul datelor, legi de învăţare şi condiţii de stabilitate). **Anexa 3** ("Alte modele de perceptron simplu") şi **anexa 4** ("Funcţii de activare ale perceptronului simplu") completează informaţiile din prima parte a cap. 3. Cea de-a doua parte a capitolului, intitulată "Structuri neurale artificiale 'feedforward' ", este dedicată abordării unor aspecte teoretice legate de învăţare, generalizare şi reducerea complexităţii modelului neural "feedforward", necesare în capitolele următoare.

Capitolul 4 ("Metode de proiectare a structurilor neurale 'feedforward' ") reprezintă o sinteză a metodelor de proiectare a structurilor neurale "feedforward", pornind de la ideea că, realizarea aplicaţiilor RNA presupune alegerea structurii şi dimensiunilor acestora. Nu există metode de proiectare general valabile, dependenţa de aplicaţie fiind subliniată. Sunt prezentate metode ontogenice (de modificare a dimensiunilor) empirice, metode bazate pe criterii statistice, metode constructive şi "distructive". In finalul capitolului sunt incluse metode mixte de proiectare, printre care o metodă secvenţială propusă de autoare, aplicată pentru proiectarea unui filtru neural pentru semnal bidimensional. Sunt prezentate comparaţii calitative ale metodelor amintite, precum şi criterii de alegere ale acestora.

Capitolul 5 ("Proiectarea structurilor neurale 'feedforward' cu constrângeri de simetrie") şi **capitolul 6** ("Proiectarea structurilor neurale 'feedforward' cu funcţii cost diferite de norma L_2 a erorii"), ca şi **anexele 5** şi **6** ("Interpretarea în planul complex a rezultatelor învăţării reţelei neurale 'feedforward' ") conţin contribuţii originale la proiectarea reţelelor neurale "feedforward". Sunt propuse metode şi algoritmi de proiectare, sunt deduse relaţiile matematice şi sunt evaluaţi indicatorii de performanţă ai structurilor rezultate.

Capitolul 7 ("Observaţii finale, concluzii şi perspective") sintetizează principalele contribuţii teoretice şi experimentale ale lucrării, formulează concluzii şi identifică direcţiile de studiu şi perspectivele. **Ultimul capitol** ("Bibliografie") conţine referinţele bibliografice ale lucrării şi este urmat de **rezumatul** lucrării în limba engleză, care prezintă succint conţinutul lucrării.

Scanned with OKEN Scanner

2 STADIUL ACTUAL AL DEZVOLTĂRII STRUCTURILOR ŞI APLICAŢIILOR REŢELELOR NEURALE ARTIFICIALE

Pentru a descrie stadiul actual al dezvoltării structurilor şi aplicaţiilor RNA, este necesară plasarea acestuia în contextul dezvoltării istorice a domeniului, precum şi în perspectiva evoluţiilor sale viitoare. Succesiunea trecut – prezent – viitor descrie dinamica apariţiei noilor modele neurale şi aplicaţiilor acestora.

2.1 TRECUT (SCURT ISTORIC)

2.1.1 Scurt istoric al modelelor neurale artificiale

Explicaţii teoretice legate de creier şi funcţionarea sa datează încă din antichitate [V200, W10]. Modelele artificiale de structuri neurale au apărut mult mai târziu decât cele biologice, iar evoluţia acestora poate fi împărţită în mai multe etape. Două dintre periodizările caracterizate prin plasticitate sunt prezentate în lucrările [M17] şi [E5], iar principalele momente în tab. 2.1. In continuare vom utiliza denumirile perioadelor propuse de Eberhart [E5].

2.1.1.1 Epoca lui Camelot ("Dinozaurii", 1890 – 1960)

In anul 1890, psihologul american William James a formulat pentru prima oară câteva dintre principiile învăţării corelate şi memoriei asociative. El aproape a prezis ideea dependenţei activităţii neuronului de suma valorilor semnalelor sale de intrare. După încă o jumătate de secol, McCulloch şi Pitts au sintetizat informaţiile biologice disponibile, referitoare la neuron şi au descris modelul artificial cunoscut sub numele de "neuronul McCulloch – Pitts" [H11]. Contribuţia lor este valoroasă mai ales prin faptul că, autorii au demonstrat reprezentabilitatea oricărei expresii logice finite, cu ajutorul reţelelor alcătuite din neuronii McCulloch – Pitts. De asemenea, au utilizat o arhitectură puternic paralelă şi au deschis drumul dezvoltării modelelor neurale. Există însă şi alte opinii, care afirmă că aceştia au contribuit nesemnificativ la evoluţia ulterioară a ştiinţelor neurale [M17].

Donald O. Hebb, în cartea sa, "The Organization of Behavior" (1949), defineşte metoda de actualizare a ponderilor cunoscută azi drept "hebbiană". El a fost printre primii care au utilizat termenul "conexionism".

Scanned with OKEN Scanner

Tab. 2.1 Principalele momente în apariţia modelelor neurale artificiale[4]

Modelul neural	Anul apariţiei	Inventator	Avantajele modelului	Dezavantajele modelului
Perceptron simplu	1943	McCulloch şi Pitts	Model simplu	Realizează numai clasificare liniară
Învăţare hebbiană*	1949	Hebb	Plauzibilitate biologică	
*	1951	Minsky şi Papert		Subliniază limitele perceptronului
Perceptron	1957	Rosenblatt	Model simplu	Realizează numai clasificare liniară
ADALINE, MADALINE	1960	B. Widrow	Rapid, uşor de implementat VLSI	Relaţie liniară între intrare şi ieşire, deci numai clasificare liniară
Reţea "Brain–State–in–a–Box" (BSB)	1977	J. Anderson	Performanţe uneori mai bune decât reţeaua Hopfield	Studiat insuficient din punct de vedere al performanţelor şi aplicaţiilor
*	1977	J. McClelland, D. Rumelhart şi grupul PDP	Începutul cercetării interdisciplinare	
Reţeaua "Learning Vector Quantization" (LVQ)	1981	T. Kohonen	Model capabil să se auto–organizeze. Funcţionare rapidă după ce învăţarea s-a încheiat	Învăţare lentă. Selectarea numărului vectorilor de învăţare şi a timpului necesar acesteia este încă nerezolvată
Harta auto – organizabilă, cu reprezentarea trăsătu – rilor: "Self Organising Map" (SOM)	1981	T. Kohonen	Model capabil să realizeze auto–organizarea datelor de intrare şi ordonarea sugestivă a reprezentării acestora	Învăţare lentă. Problema selectării numărului vectorilor pentru învăţare şi a timpului necesar acesteia nu este încă deplin rezolvată
Reţeaua "Counterpropagation"	1982	Hecht – Nielsen	Model de tip "feedforward" – "feed back"	Structură complexă
Reţeaua Hopfield	1982	J. Hopfield	Model simplu Stabilitatea demonstrată Implementare VLSI simplă	Modelul discret are ponderi fixe. deci este incapabil să înveţe noi stări. Capacitate de stocare redusă

[4] S-au marcat cu * momente importante pentru dezvoltarea modelelor neurale, deşi nici un model nou nu a apărut.

Scanned with OKEN Scanner

Tab. 2.1 Principalele momente în apariţia modelelor neurale artificiale (cont.)

Modelul neural	Anul apariţiei	Inventator	Avantajele modelului	Dezavantajele modelului
Neocognitron	1975 – – 1982	K. Fukushima	Performanţe bune în recunoaşterea formelor, invariantă la translaţie şi rotaţie	Structură complexă, multe elemente de prelucrare. Scalarea datelor pentru probleme reale este încă nerezolvată
Perceptron multistrat şi algoritmul BKP	1974 – – 1986	P.J.Werbos, D. Rumelhart	Operează rapid în faza de test. Perfomanţe foarte bune în aplicaţii	Învăţare lentă. Convergenţa nu este garantată
ART ("Adaptive Resonance Theory")	1983	G. Carpenter şi S. Grossberg	Poate învăţa cu uşurinţă noi patternuri şi le poate apoi recupera	Categoriile (clasele) se pot modifica în timpul învăţării
Maşina Boltzmann, Maşina Cauchy	1984, 1986	G. Hinton, T. Sejnowski, H. Szu	Model capabil să formeze reprezentări optimale ale trăsăturilor existente în patternuri	Timp de învăţare foarte mare pentru maşina Boltzmann şi mai redus pentru maşina Cauchy
Memoria asociativă bidirecţională (BAM)	1986	Bart Kosko	Model cu arhitectură, lege de învăţare şi dinamică simple. Structură stabilă	Capacitate de stocare redusă. Acurateţe redusă în recuperarea patternurilor stocate
Structuri recurente	1987	Almeida, Pineda	Perfomanţe deosebite în clasificarea informaţiei variabile în timp	Complexitate mare, învăţarea şi optimizarea sunt dificile
Reţea "Time – Delay Neural Network" (TDNN)	1987	D.W. Tank, J.J. Hopfield	Perfomanţe echivalente celor mai bune metode convenţionale. Viteză ridicată	Dimensiunea ferestrei temporale de obicei fixă. Performanţa depinde de scalarea semnalelor de intrare
Reţea "Functional – Link Network"	1988	Y.H. Pao	Conţine numai stratul de intrare şi de ieşire. Învăţarea este rapidă	Alegerea funcţiilor pentru legăturile funcţionale este euristică
Reţea cu funcţii de bază radială ("Radial Basis Functions Net", RBF)	1987, 1988	Mai mulţi cercetători	Reţeaua RBF cu un strat ascuns de neuroni RBF se comportă echivalent reţelei perceptron multistrat cu două straturi ascunse	Învăţare uneori îndelungată
Număr mare de variante ale modelelor anterioare	1989 – 1996	Mai mulţi cercetători	Modificări ale modelelor pentru obţinerea unor performanţe mai bune	Diversificare "orizontală"

Scanned with OKEN Scanner

In anul 1958, Frank Rosenblatt a utilizat modelul biologic al celulelor din retină, pentru a defini structura numită "perceptron", compusă dintr-un strat de perceptroni simpli [E5, H11]. Lucrarea lui Rosenblatt a deschis drumul cercetării atât în direcţia învăţării supervizate, cât şi nesupervizate. Bernard Widrow şi Marcian Hoff au introdus în 1960 modelul "ADALINE" ("ADAptive LINEar"), cu un algoritm de învăţare supervizată.

2.1.1.2 Perioada întunecată (1969 – 1980)

Lucrarea "Perceptrons" publicată în anul 1969 de Marvin Minsky şi Seymour Papert a constituit momentul intrării într-o perioadă dificilă a dezvoltării modelelor neurale. Cartea sublinia limitele şi dezavantajele perceptronului. Arătând că, acesta este capabil să realizeze numai clasificare liniară şi nu poate rezolva nici probleme neliniare simple, ca problema SAU – EXCLUSIV (XOR), Minsky şi Papert afirmau: "intuim că, extensia [la perceptron multistrat, cu straturi ascunse] este sterilă". Deşi mai târziu, după anul 1986, se va dovedi că această predicţie a fost o greşeală, Papert va continua să susţină punctul de vedere al lucrării iniţiale. Caracterizată mai târziu de Anderson şi Rosenfeld [M17] drept o lucrare "... strălucitoare ... scrisă simplu şi clar, într-un stil elegant şi neformal", cartea "Perceptrons" a avut în momentul apariţiei sale, rolul de a bloca finanţarea cercetării în domeniul RNA. Aceasta nu a însemnat şi renunţarea celor ce studiau problemele legate de modelele neurale, dar nici un proiect de anvergură nu a putut fi iniţiat timp de mai mulţi ani.

Anul 1972 a marcat publicarea în paralel, a rezultatelor de cercetare în domeniul neural de către Teuvo Kohonen, de la universitatea Helsinki University of Technology din Finlanda şi totodată, de către James Anderson, neurofiziolog american. Primul a realizat o "memorie asociativă", iar cel de-al doilea o "memorie interactivă", modelele având arhitectura, algoritmii de învăţare şi functiile de transfer aproape identice. Rezultatele sunt remarcabile şi prin faptul că, cercetătorii au lucrat independent.

In aceeaşi perioadă, Stephen Grossberg şi Gail Carpenter au creat teoria rezonanţei adaptive (ART), iar Kunihiko Fukushima a dezvoltat o serie de arhitecturi RNA, dintre care cele mai cunoscute sunt cognitronul şi neocognitronul. Trăsăturile comune ale acestor modele sunt plauzibilitatea biologică şi performanţele bune în recunoaşterea formelor. Tot în anul 1972, Shun – ichi Amari a extins studiile sale în domeniul teoriei învăţării, la modelele auto – organizabile. In acelaşi domeniu, A. Henry Klopf a propus o lege de învăţare inspirată din biologie, bazată pe ipoteza că, fiecare neuron este "hedonist" prin adaptarea pe care o realizează pentru a maximiza eficienţa sinapselor sale excitatoare atunci când are loc activarea (fenomen de plăcere) şi de a minimiza eficienţa sinapselor sale inhibitoare (fenomen de durere) [M17]. Legea de învăţare bazată pe această ipoteză a fost numită "drive – reinforcement

Scanned with OKEN Scanner

learning". Un alt model apărut în anii '70 şi inspirat din neurofiziologie a fost reţeaua "Brain–State–in–a–Box" (BSB), propusă de James Anderson [H11].

2.1.1.3 Renaşterea (1982 – 1986)

Perioada imediat următoare anului 1982 s-a caracterizat printr-o modificare a atitudinii generale faţă de studiile legate de RNA. La aceasta au contribuit rezultatele obţinute de cercetători şi crearea a două modele, care au înregistrat ulterior cel mai mare număr de aplicaţii dintre reţelele neurale artificiale.

Fizicianul John J. Hopfield a propus în anul 1982 modelul numit reţeaua Hopfield. Ideea sa novatoare a fost legată de formularea definiţiei energiei reţelei neurale, precum şi a caracterizării modelului pe baza acesteia. Pe de altă parte, a devenit evidentă în scurt timp şi limita reţelei, legată de capacitatea de stocare redusă. Simplitatea sa a determinat însă apariţia primei implementări hardware în 1984, realizată de AT & T Bell Laboratories [K8].

Cel de-al doilea moment care a marcat perioada de "renaştere" a interesului pentru domeniul RNA a fost publicarea în anul 1986, de către James McClelland, David E. Rumelhart şi grupul de cercetare PDP ("Parallel Distributed Processing"), a lucrării de referinţă "Parallel Distributed Processing", considerată una dintre "bibliile tehnice" ale domeniului. In aceeaşi perioadă a apărut modelul perceptron multistrat (RPM) cu algoritmul de propagare inversă ("Backpropagation Algorithm" — BKP) publicat de Paul Werbos şi Dave Parker. Aplicaţia propusă de Werbos a fost predicţia datelor din domeniul economic, dar în anii care au urmat modelul a fost aplicat pentru rezolvarea unui mare număr de probleme.

2.1.1.4 Epoca neoconexionistă (din 1987 până în prezent)

Cowan şi Sharp au introdus denumirea de "neoconexionism" pentru realizările din domeniul RNA, în această perioadă. Intervalul de timp menţionat se caracterizează atât prin crearea unor noi modele (arhitecturi), algoritmi de învăţare, dar şi aplicarea acestora pentru rezolvarea unor probleme noi.

In anul 1988, agenţia americană "Defense Advanced Research Projects Agency" (DARPA) a iniţiat un proiect amplu de cercetare în domeniul reţelelor neurale, recomandând totodată realizarea unui efort financiar de 50 milioane dolari, pe o durată de 8 ani, pentru susţinerea acestuia. Conducerea DARPA a alocat 33 milioane de dolari pentru un program cu durata de 2 ani şi jumătate, ale cărui obiective şi concluzii sunt prezentate în tab. 2.2. Toate cele trei scopuri iniţiale ale programului au fost realizate, stabilindu-se şi obiectivele următoare anului 1992 [J17].

Scanned with OKEN Scanner

Aproape simultan derulării proiectului DARPA, în Japonia se afla în curs de desfăşurare proiectul guvernamental "Cea de-a cincea generaţie de calculatoare", la care ne vom referi ulterior.

Curând după iniţierea proiectului DARPA, au început să apară o serie de proiecte şi în mediul universitar, precum şi cursuri noi de reţele neurale. Conferinţele în domeniu, revistele de specialitate şi fuzionarea unora dintre manifestările existente (de exemplu, conferinţele anuale ale IEEE şi ale International Society on Neural Networks au fuzionat în 1989, denumirea evenimentului fiind International Joint Conference on Neural Networks) au contribuit la intensificarea activităţii de cercetare în domeniu. Au urmat o serie de proiecte europene de amploare diferită, în perioada 1990 – 1991, iar din anul 1994 multe dintre acestea au devenit componente ale proiectelor "de excelenţă" de reală anvergură, cu participare universitară şi industrială. Existenţa preocupărilor susţinute de fonduri financiare importante, la nivel guvernamental sau local, a avut un impact decisiv asupra dezvoltării generale a cercetării în domeniul RNA.

Tab 2.2 Obiectivele şi concluziile proiectului DARPA

Obiectivele proiectului DARPA	Concluzii
☑ Măsurări ale performanţelor folosind baze de date standard pentru recunoaşterea formelor	RNA sunt utile în clasificarea formelor
☑ Dezvoltarea tehnologică, mai ales în domeniul circuitelor digitale	Prima generaţie de circuite neurale digitale
☑ Caracterizarea teoretică a arhitecturii neurale	Există aspecte dificile şi dezavantaje ale RNA

Notă: Semnul ☑ marchează obiectivele realizate

2.1.2 Scurt istoric al aplicaţiilor modelelor neurale artificiale în prelucrarea digitală a semnalelor

Deşi s-au dezvoltat independent, domeniul prelucrării digitale a semnalelor şi RNA sunt strâns legate. In anul 1949, la numai câţiva ani după ce McCulloch şi Pitts au introdus structura de tip perceptron simplu, Wiener a propus un model general de filtru neliniar, prin particularizarea căruia se pot obţine multe dintre structurile neurale artificiale de tip "feedforward". La numai doi ani după ce Widrow şi Hoff au formulat caracteristicile modelului neural ADALINE, Schetzen (pentru modelul Wiener) şi Rosenblatt (pentru perceptron) au propus algoritmi iterativi de învăţare. Trăsătura comună a modelelor menţionate este

Scanned with OKEN Scanner

adaptabilitatea, de aceea este firesc faptul că, primele aplicaţii ale RNA în PDS au fost cele de filtrare adaptivă.

In perioada de stagnare a dezvoltării RNA, domeniul PDS şi-a continuat evoluţia. Complexitatea crescândă a modelelor a pus în evidenţă necesitatea găsirii unor structuri mai performante pentru aplicaţiile neliniare. Joncţiunea dintre domeniile RNA şi PDS a avut loc mai ales în ultimul deceniu şi jumătate. S-au pus în evidenţă rezultatele obţinute, avantajele şi dezavantajele modelelor neurale artificiale în filtrare, estimare, detecţie, predicţie, etc.

In tab. 2.3 este prezentat un scurt istoric al aplicării diferitelor modele neurale în prelucrarea semnalelor, iar în tab. 2.4, al implementărilor acestora. In ambele cazuri, anul marchează prima aplicaţie a modelului neural respectiv, înregistrată în referinţele bibliografice.

Tab 2.3 Apariţia primelor aplicaţii ale diferitelor modele neurale

Aplicaţii	Perc. simplu	ADA LINE	RPM cu BKP	Maşina Boltz–mann	LVQ	Hopfield cont.	ART	BAM	BSB
Recunoaşterea formelor în general	1960	1962	1988	1986		1987			1988
Recunoaşterea caracterelor			1988						
Prelucrarea semnalelor în general			1987			1986			
Predicţie / Optimizare			1986						
Eliminarea zgomotului		1975							
Compresia datelor					1988				
Prelucrarea imaginilor	1962	1988	1988	1985		1987	1986	1986	1987
Prelucrarea semnalului vocal	1962		1986	1986	1984		1986		
Prelucrare temporală		1981	1988				1988		
Optimizare				1987	1988	1986			
Diagnostic			1988				1987	1988	1987
Control	1988	1964	1987		1984	1987	1987		
Prelucrarea cunoştinţelor	1986		1986	1986			1988		1981
Experimente de psihologie							1982		1978
Realitate virtuală			1994						

Scanned with OKEN Scanner

Tab 2.4 Apariţia primelor implementări ale diferitelor modele neurale

Implementare	RPM cu BKP	Maşina Boltzmann	LVQ	Hopfield cont.	ART	BAM
Circuit integrat	1988	1986	1988	1986	1988	
Procesor / Coprocesor ataşat	1987	1988	1987	1987	1987	1987
Procesor de tip "bus – oriented"	1987		1987	1987		
Optică / Electro – optică	1987	1987		1987		1987
Calculatoare puternic paralele				1987		

2.2 PREZENT (STADIUL ACTUAL)

2.2.1 Stadiul actual al dezvoltării modelelor neurale artificiale

In prezent există o varietate de modele neurale artificiale, cu diverse variante, care au performanţe îmbunătăţite. O prezentare sugestivă a stadiului actual al modelelor neurale este realizată de Maren, Lee şi Sheu (fig. 2.1) [M17, L41].

Fig. 2.2 prezintă tipuri de implementări RNA existente: software, hardware (analogice, digitale, optice şi hibride). Se poate constata orientarea diferitelor companii, la un moment dat, spre o anumită tehnologie. De exemplu, primele mari companii japoneze producătoare de hardware neural utilizează: hardware deja disponibil (Fujitsu), hardware neural cu circuite VLSI (Hitachi), hardware optic (Mitsubishi Electronics), iar Wacom foloseşte semnale magnetice pentru interconectarea neuronilor analogici [F10, L14, ş.a.].

Se poate identifica, de asemenea, o orientare spre cercetarea teoretică sau aplicativă şi în funcţie de zona geografică (tab. 2.5). Semnul ☑ marchează, prin repetare, preocuparea pentru un anumit tip de cercetare.

Tab 2.5 Priorităţi teoretice, software şi hardware în domeniul neural, în funcţie de diferitele zone geografice

Zona geografică	Studii teoretice	Software	Hardware digital / analog	Hardware optic şi hibrid
Europa	☑☑☑	☑☑	☑☑	☑
Japonia	☑☑	☑	☑☑	☑☑☑
Statele Unite	☑☑	☑☑☑	☑☑☑	☑

Scanned with OKEN Scanner

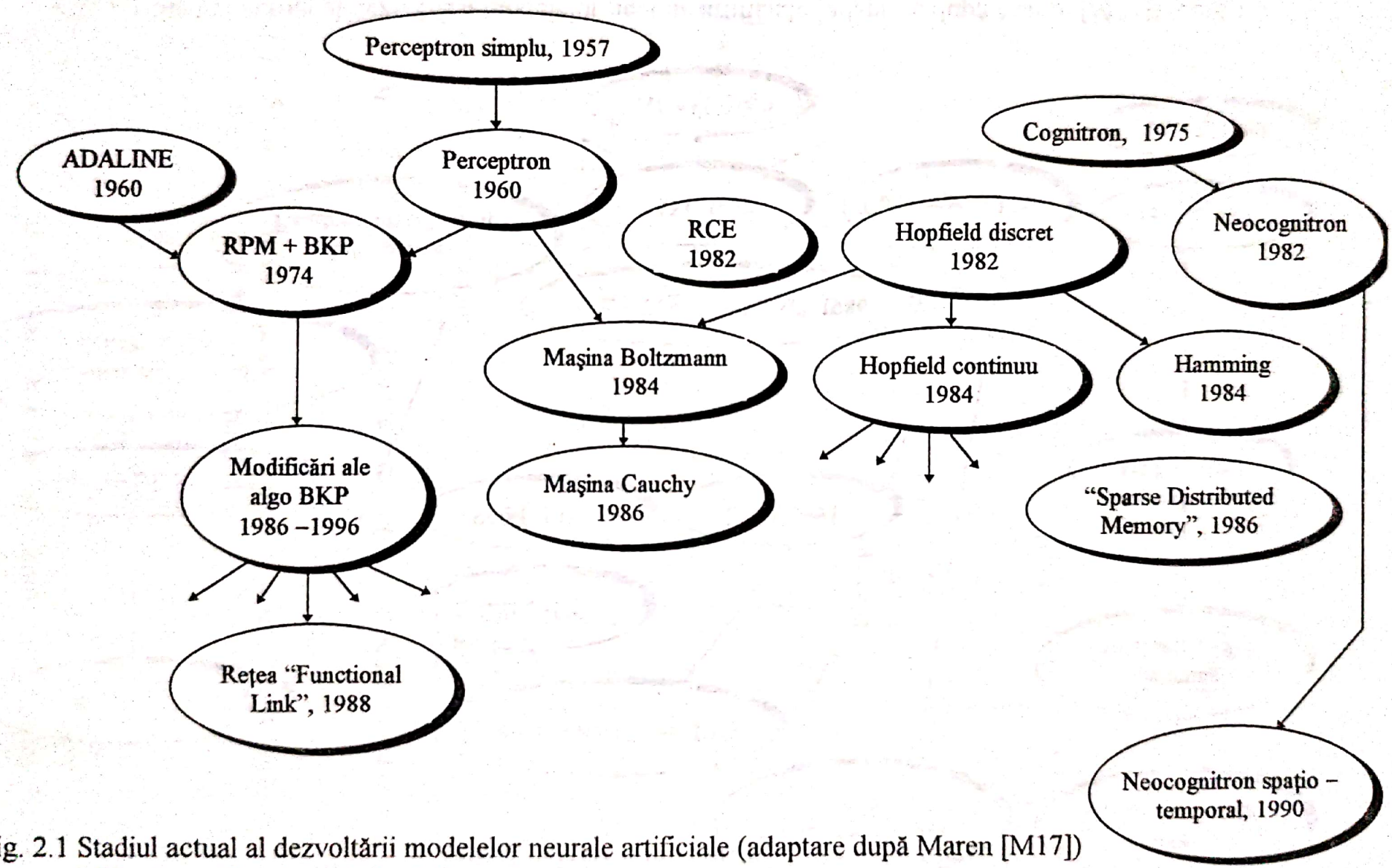


Fig. 2.1 Stadiul actual al dezvoltării modelelor neurale artificiale (adaptare după Maren [M17])
RCE = Reţea "Restricted Coulomb Energy"

Scanned with OKEN Scanner

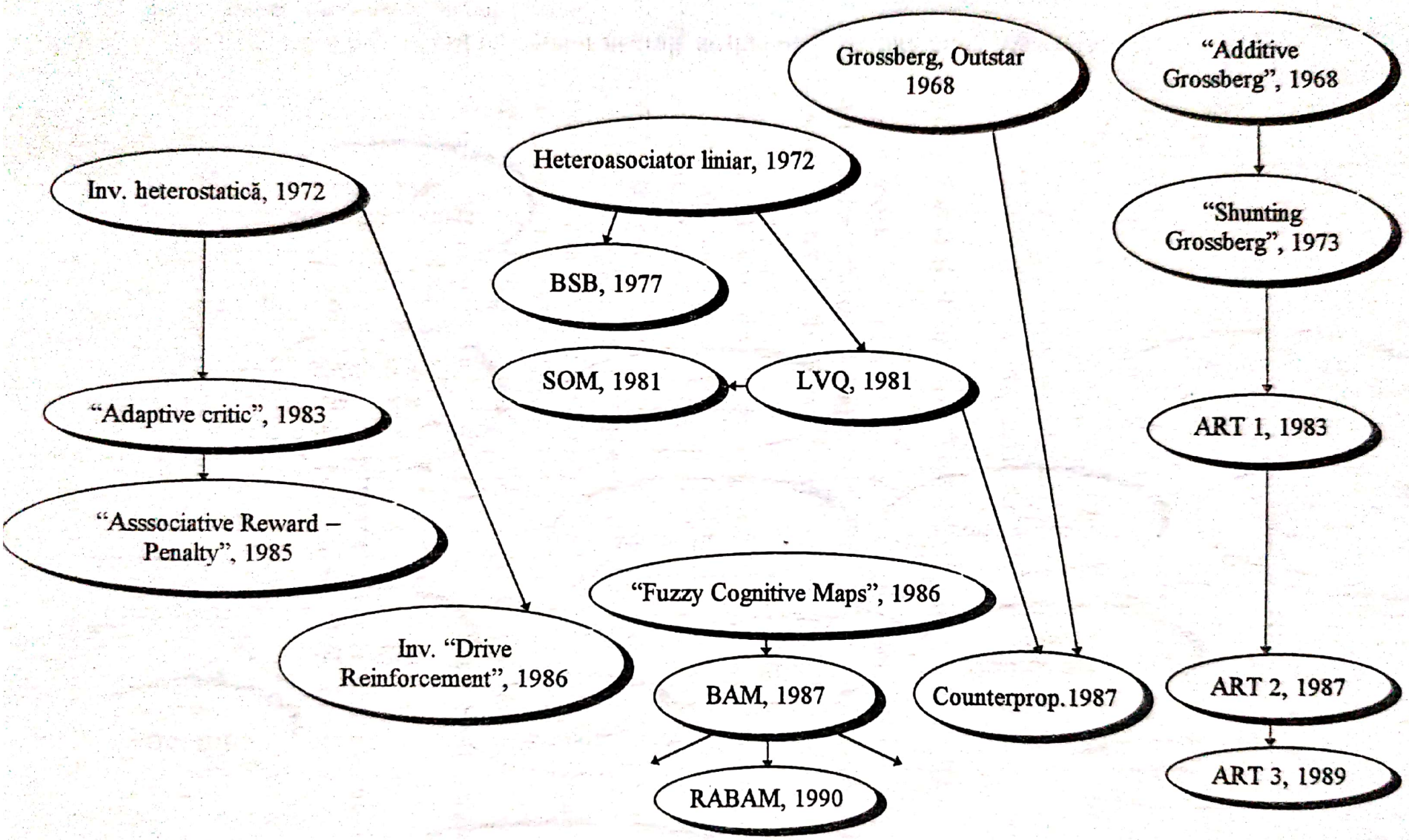


Fig. 2.1 Stadiul actual al dezvoltării modelelor neurale artificiale (adaptare după Maren [M17]) (cont.)

Scanned with OKEN Scanner

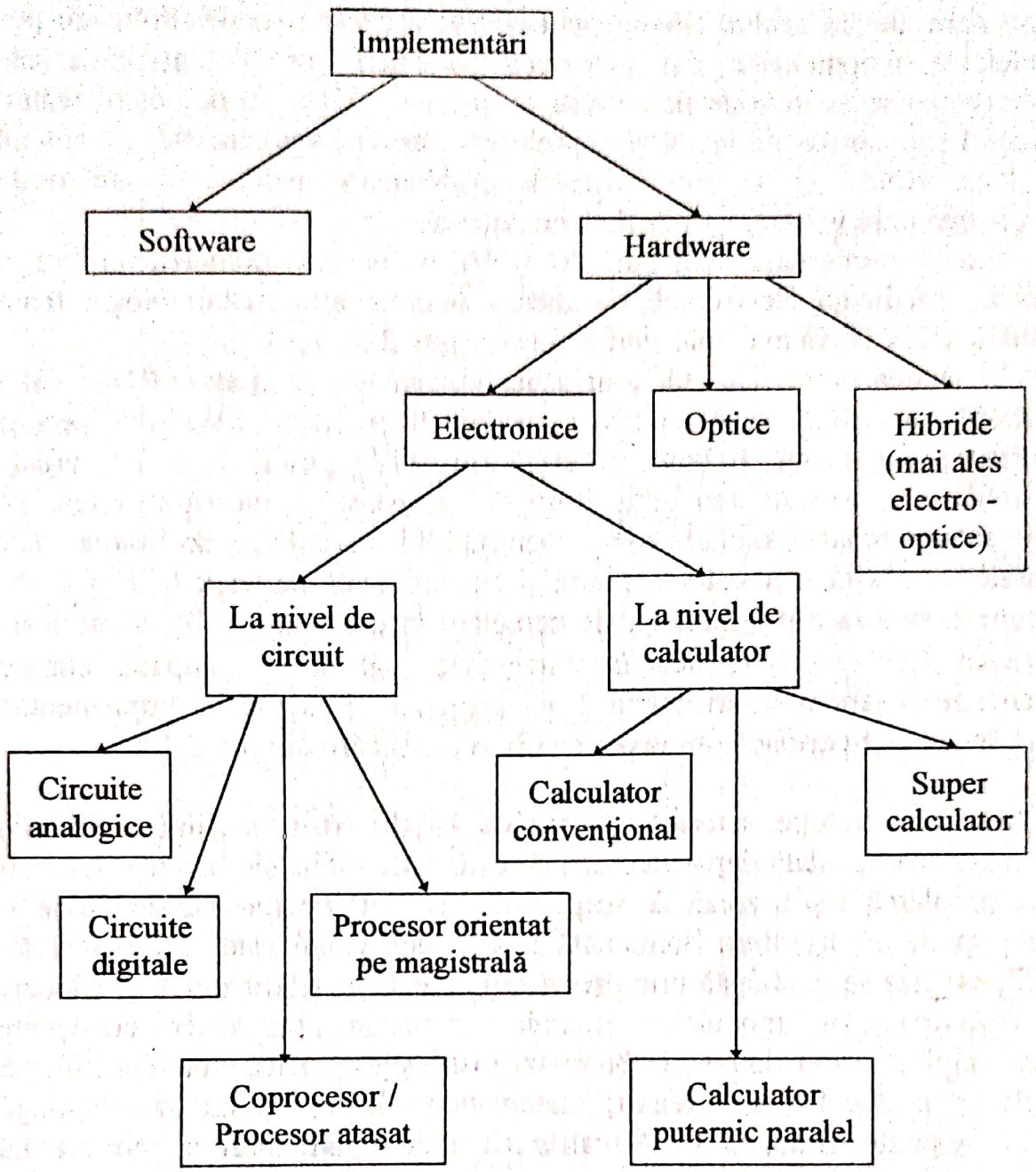


Fig. 2.2 Tipuri de implementări ale modelelor neurale artificiale

2.2.2 Stadiul actual al aplicaţiilor modelelor neurale artificiale în prelucrarea digitală a semnalelor

Dinamica dezvoltării aplicaţiilor RNA în PDS reflectă interesul cercetătorilor din domenii conexe. Neurofiziologii utilizează RNA pentru a descrie funcţiile de nivel mediu ale creierului (exemplu: memorarea). Cercetătorii din domeniul ştiinţelor cunoaşterii ("cognitive science") consideră RNA un instrument posibil de descriere a gândirii şi conştiinţei (funcţii

Scanned with OKEN Scanner

superioare ale creierului). Fizicienii folosesc reţelele neurale artificiale pentru modelarea fenomenelor din mecanica statistică, iar biologii încearcă să interpreteze secvenţele de nucleotide cu ajutorul RNA. In domeniul tehnic se studiază caracteristicile învăţării şi prelucrării informaţiei care nu este simbolică pe baza RNA, în scopul rezolvării problemelor neliniare, care necesită procesarea unui volum mare de date cu zgomot.

Există numeroase aplicaţii ale RNA în diferite domenii, printre care amintim: medicină, electronică, robotică, economie, artă, neurobiologie, realitate virtuală, etc. Câteva exemple sunt incluse în tab. 2.6.

Dinamica puternică, atât a apariţiei variantelor de modele RNA, cât şi a aplicaţiilor acestora, a fost sugestiv caracterizată de Bernard Widrow: "aşa cum, în fiecare dimineaţă, trebuie să urmăresc ştirile pentru a fi la curent cu schimbările guvernelor din Estul Europei, tot astfel trebuie să-mi citesc poşta dimineaţa (care acum include email), pentru a fi la curent cu dezvoltarea reţelelor neurale". Această idee este susţinută şi exemplificată de anexele 1 şi 2. Prima conţine o sinteză a programelor de cercetare în domeniul RNA, identificate în perioada 1990 – 1996, atât în universităţi, cât şi în companii europene, americane şi japoneze. In anexa 2 am prezentat un sumar al implementărilor RNA în aceeaşi perioadă, ordonate conform clasificării din fig. 2.2.

După o evoluţie istorică ce prezintă variaţii bruşte ale interesului pentru domeniul RNA, odată depăşită etapa de exagerare ("reţelele neurale pot rezolva orice problemă") şi a zonei de scepticism ("reţelele neurale nu sunt utile"), se poate spune că trăsătura dominantă a stadiului actual este aceea de realism [K27]. Acesta se manifestă prin precizarea corectă a performanţelor, avantajelor şi dezavantajelor modelelor neurale artificiale, pe baza cunoştinţelor interdisciplinare din domeniile biologiei, psihologiei, calculatoarelor, ingineriei electrice, prelucrării semnalelor, matematicii, fizicii, opticii şi tehnologiilor VLSI. Se poate afirma că, RNA realizează pe de o parte lărgirea unor orizonturi deja existente şi deschiderea de noi perspective de cercetare.

2.3 VIITOR (PERSPECTIVE)

O serie de proiecte de amploare în domeniul reţelelor neurale se află în curs de desfăşurare, durata lor variind de la câţiva ani la un deceniu. Printre acestea se numără proiectele japoneze şi cele ale agenţiei americane DARPA.

Proiectul celei de-a cincea generaţii de calculatoare (fig. 2.3), marcat de dezvoltarea ingineriei cunoaşterii ("knowledge engineering"), a fost succedat de proiectul celei de-a şasea generaţii, aflat în desfăşurare. Acesta urmăreşte realizarea sistemelor PINK ("Psychological / Intelligent / Neural / Knowledge Engineering") [S11].

Scanned with OKEN Scanner

Tab. 2.6 Exemple de aplicaţii ale reţelelor neurale artificiale în diferite domenii

Domeniul	Exemple de aplicaţii
Medicină	Prelucrarea semnalelor medicale (EEG, ECG, etc.)
	Clasificarea în domeniul psihiatric
	Sisteme de clasificare a durerilor de spate, a afecţiunilor cardiace, etc.
	Clasificarea ţipetelor sugarilor
	Detecţia celulelor canceroase
	Predicţia numărului de globule roşii în sânge
Industrie în general	Controlul de calitate al produselor (identificarea defectelor)
	Echipamente de control pentru navigaţie autonomă, planificarea rutei automobilistice cu evitarea obstacolelor
	Modelarea neliniarităţilor în controlul nivelului pH al lichidelor, în industria chimică
	Predicţia consumului de apă în oraşul Paris
	Predicţia consumului de energie electrică în oraşul Tokio
	Analiza de calitate a apei potabile
	Predicţia smog–ului
Economie şi afaceri	Predicţia necesarului de locuri în avion
	Predicţia ratei dobânzilor
	Selecţia angajaţilor
Artă	Compozitor neural
	Pictor neural
Diverse	Nas neural
	Alegerea câştigătorilor la cursele hipice
	Predicţia rezultatului în fotbal, "backgammon", bridge, Kai – fu Lee, GO, Sanjoy Mahajan, etc.
	Selecţia ţintelor de investigaţie în anchetele penale
	Identificarea şi recunoaşterea ţintelor militare
	Controlul semnăturii pe cecurile de bancă
	Recunoaşterea numărului de la maşină (garaje, în Germania)
	Detecţia materialelor explizibile în bagajele călătorilor (în funcţiune în aeroporturile Heathrow, Dallas, ş.a.)
	Modelarea creşterii recoltelor agricole
	Controlul dimensiunii şi culorii merelor pentru export în Japonia

In sistemele de calcul din generaţia a şasea, modelele neurale sunt plasate pe nivelul al doilea, dintre cele trei nivele de inteligenţă artificială (fig. 2.4):

Scanned with OKEN Scanner

❶ inteligenţa artificială de bază, similară celei din sistemele generaţiei a cincea; ❷ inteligenţa neurală (BBI, "Brain – Behavior Intelligence"); ❸ inteligenţa moleculară (MI, "Molecular Intelligence"), la nivelul citoscheletului celulei vii, ca nivel molecular de cunoaştere şi prelucrare a informaţiei.

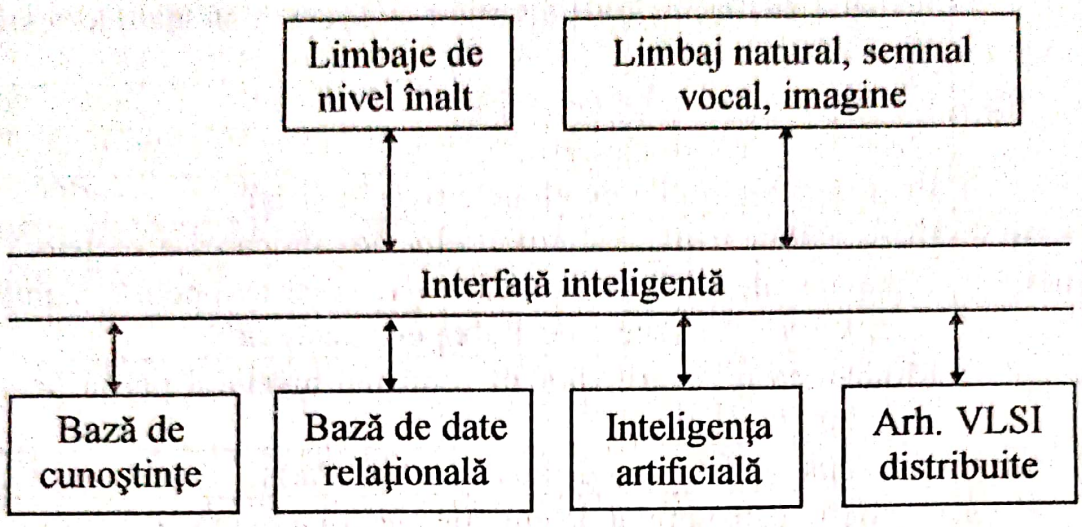


Fig. 2.3 Sistemele de calcul din generaţia a cincea

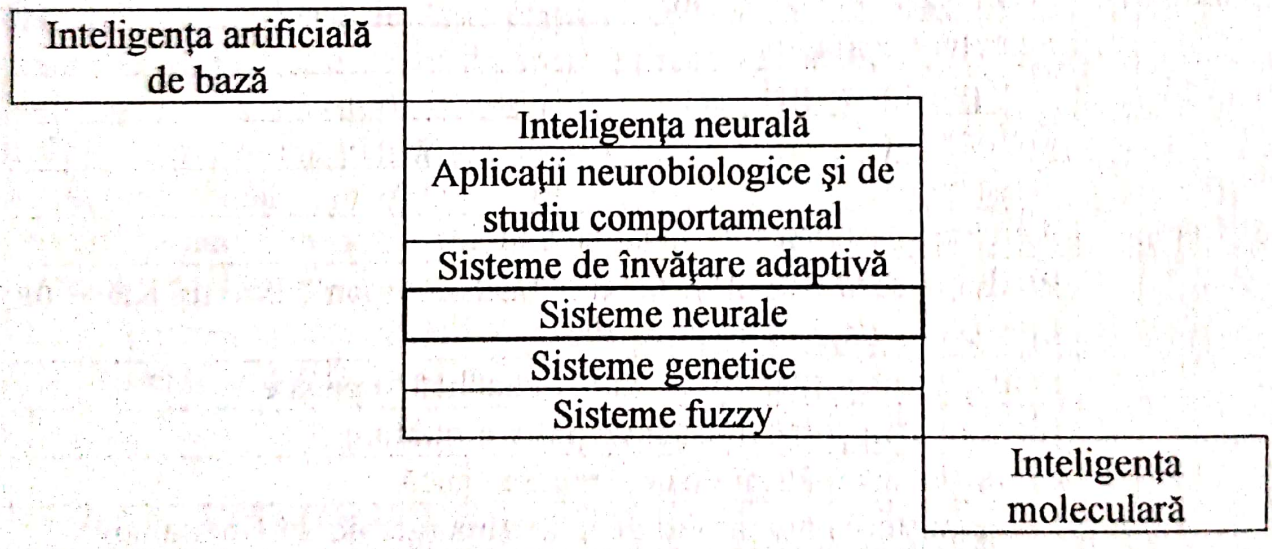


Fig. 2.4 Nivelele de inteligenţă artificială în sistemele de calcul din gen. a şasea

O altă dihotomie a sistemelor din generaţia a şasea subliniază existenţa abordărilor simbolică şi conexionistă. Acestea diferă prin nivelul la care explică procesele cognitive. Prima variantă descrie inteligenţa umană pe baza simbolurilor şi regulilor care operează cu simboluri. La cea de-a doua ne-am referit anterior şi se bazează pe modele constituite din populaţii de elemente de prelucrare simple (neuroni artificiali) [B14]. Sistemele ce aparţin fiecărei clase

Scanned with OKEN Scanner

menţionate sunt prezentate în tab. 2.7. Sistemele neurale sunt caracterizate prin stocarea informaţiei (cunoştinţelor) nestructurate, sub formă numerică.

Tab. 2.7 Reprezentare simbolică vs. numerică şi cunoştinţe structurate vs. nestructurate

Informaţia din punct de vedere al conţinutului	Informaţia ca reprezentare	
	Simbolică	Numerică
Structurată	Sisteme expert	Sisteme fuzzy
Nestructurată	—	Sisteme neurale

Un succesor al proiectului celei de-a cincea generaţii de calculatoare este proiectul japonez NIPT şi se estimează că va avea un impact major asupra domeniului RNA [L14]. Scopul său este acela de a implementa prelucrarea informaţiei "de tip uman" în calculatoare puternic paralele. Cele trei direcţii de cercetare ale proiectului sunt: reţele neurale, prelucrarea optică a informaţiei şi prelucrarea puternic paralelă a informaţiei. Deosebit de important este faptul că, spre deosebire de proiectele anterioare, care erau centralizate şi deschise numai participării industriei japoneze, proiectul NIPT este descentralizat şi deschis participării japoneze şi străine din mediul universitar.

Un alt proiect important iniţiat de guvernul japonez sub titlul "Human Frontier Science Program" (HFSP) are drept scop concentrarea cercetării în direcţia elucidării mecanismelor complexe ale organismelor vii. Modelele artificiale au un rol important în cadrul acestui proiect internaţional [F10].

Direcţia de cercetare şi dezvoltare a aplicaţiilor artificiale care să conducă la "reţelele neurale plauzibile biologic" ("Biologically Realistic Neural Networks") este semnalată de Whitehouse [W16] şi reprezintă un obiectiv important al proiectelor japoneze. Pornind pe de o parte, de la observaţia că, natura încearcă să găsească soluţii optime şi, pe de altă parte, că pe baza modelului neural artificial simplu nu se poate explica mecanismul complex de funcţionare a creierului uman, Whitehouse propune includerea în cadrul modelelor artificiale, a unor noi trăsături ale modelului biologic.

In timp ce proiectele japoneze se orientează spre studiul interdisciplinar şi al tehnologiilor optice şi hibride, proiectele americane au o evoluţie neaşteptată în ultimii 2 – 3 ani, predicţia fiind dificilă. După îndeplinirea obiectivelor proiectului DARPA iniţiat în 1988 şi încheiat în 1991, au fost trasate obiectivele următorului program al agenţiei americane în domeniul realizării aplicaţiilor şi al cercetării tehnologice [D205]. Anunţul pe care DARPA l-a făcut la conferinţa IJCNN'1991 ("Intl. Joint Conference on Neural Networks"), de a reduce

Scanned with OKEN Scanner

finanţarea cercetării în domeniul reţelelor neurale, începând cu anul 1992, cu circa 40%, a constituit o surpriză. Ideea formulată de DARPA a fost aceea de reducere a fondurilor alocate până la valoarea sumei investite de Comunitatea Europeană şi de Japonia în acelaşi scop. Un element important îl constituie şi faptul că, din suma totală (din care, 8 milioane dolari au fost furnizate de DARPA, 12.5 milioane dolari de Forţelor Navale Americane, 7 milioane dolari de Forţele Armate, 1.5 milioane dolari de Fondul naţional de cercetare ştiinţifică NSF — "National Science Foundation" şi 1 milion dolari de Institutul naţional de sănătate — "National Institute of Health"), numai 9% au fost alocate aplicaţiilor civile. Statele europene şi Japonia au indicat aproximativ 18% din totalul fondurilor pentru cercetare neurală, destinate aplicaţiilor civile. Prin reducerea finanţării şi alocarea fondurilor pentru aplicaţii militare, DARPA a indicat de fapt orientarea cercetării neurale pentru următorii ani. Este destul de puţin posibil ca aceasta să se modifice înaintea de terminarea proiectului, dar pe de altă parte, este dificilă realizarea predicţiei tipurilor de aplicaţii, întrucât acestea nu sunt făcute publice [J17].

❒ *Perspective*

O parte dintre predicţiile referitoare la viitorul domeniului RNA iau în considerare arhitecturile neurale şi aspectele teoretice legate de acestea [M17, S36]:

❶ Modelele neurale vor deveni un instrument esenţial de prelucrare a informaţiei.
❷ Modelele neurale vor evolua în direcţia apariţiei unoi noi interfeţe om – maşină. Urmarea va fi integrarea acestora în sisteme performante de control, educaţie şi de timp liber, care să definească o "realitate alternativă".
❸ Modelele neurale se vor perfecţiona în direcţia unor structuri plauzibile biologic pe de o parte, care vor face posibilă, pe de altă parte, o cunoaştere profundă a domeniului biologic.
❹ Lecţiile obţinute prin evaluarea limitelor modelelor neurale existente vor constitui baze de studiu pentru cercetarea în domeniul software-ului şi neurobiologiei.

❒ *Probleme nerezolvate până în prezent, legate de modelele neurale*

In domeniul teoretic neural există probleme care până în prezent au rămas nerezolvate. Printre acestea se numără [E9, L17, F37]:

Scanned with OKEN Scanner

❶ Unificarea teoriei matematice ce stă la baza RNA
❷ Proiectarea arhitecturilor optimale "feedforward"
❸ Reducerea complexităţii modelelor neurale
❹ Studiul influenţei reprezentării informaţiei şi scalării datelor, în problemele reale, asupra performanţelor modelului
❺ Studiul stabilităţii
❻ Evaluarea fiabilităţii oferite de modelelor neurale în aplicaţii specifice
❼ Diversificarea aplicaţiilor militare şi civile, bazate pe RNA
❽ Integrarea structurilor neurale în sisteme de prelucrare de mari dimensiuni
❾ Realizarea standardizării (terminologie, notaţii şi "benchmark")

Este încă prea devreme să descriem efectul pe termen lung al reţelelor neurale asupra neurobiologiei şi neuroştiinţelor experimentale [H19]. Este cert că, suntem încă departe de structurile de tip DWIM ("Do What I Mean"[5]). După cum, şi speculaţiile privind producerea într-o zi a maşinilor capabile să perceapă asemenea omului, să gândească şi să se mişte aidoma lui, pe baza structurilor neurale artificiale ("alimentate de terminologia antropomorfică, inspiraţia biologică, etc." [Z2]), nu se referă la viitorul apropiat [H7].

Există lucrări care au curajul realizării unor predicţii privind evoluţia RNA în următoarele decenii. Acestea oscilează de la nuanţe nihiliste spre cele optimiste. Câteva aspecte sunt însă certe: modelele neurale artificiale trebuie integrate în sistemele utile existente, pentru a putea fi folosite în aplicaţii de mari dimensiuni, şi, de asemenea, trebuie făcuţi paşi importanţi spre unificarea teoriilor, metodelor şi abordărilor existente.

2.4 REZUMATUL CAPITOLULUI

In cadrul acestui capitol am prezentat stadiul actual al modelelor de RNA şi aplicaţiilor acestora în prelucrarea digitală a semnalelor, din perspectiva trecut – prezent – viitor. După un scurt istoric al domeniului, am subliniat principalele tendinţe actuale şi de viitor, precum şi problemele nerezolvate, care concentrează atenţia cercetătorilor din numeroase domenii conexe neuroinformaticii.

[5] Fă ceea ce vreau să spun

Scanned with OKEN Scanner

3 BAZE TEORETICE ALE ABORDĂRII MODELELOR NEURALE ARTIFICIALE

3.1 CARACTERISTICILE MODELELOR NEURALE ARTIFICIALE

Există diferite tipuri de rețele neurale (fig. 3.1). Dintre acestea, ne vom referi la modelele neurale artificiale, definite de următoarele elemente: caracteristicile nodului (perceptron simplu), topologia rețelei și legea de învățare [B9, Z2, ș.a.]. Acestora li se adaugă criteriul de cost și condițiile de stabilitate. Există și alte taxonomii neurale, printre care cea datorată lui Fiesler [F37].

Din punct de vedere structural, se poate considera că modelul neural este ierarhizat la următoarele nivele: micro – structură, mezo – structură și macro – structură. La nivelul micro – structurii se află perceptronul simplu sau neuronul artificial, la cel al mezo – structurii rețelele neurale ca topologii de interconectare a perceptronilor simpli, iar macro – structura se realizează prin luarea în considerare a arhitecturilor compuse din structuri de rețele neurale [M17].

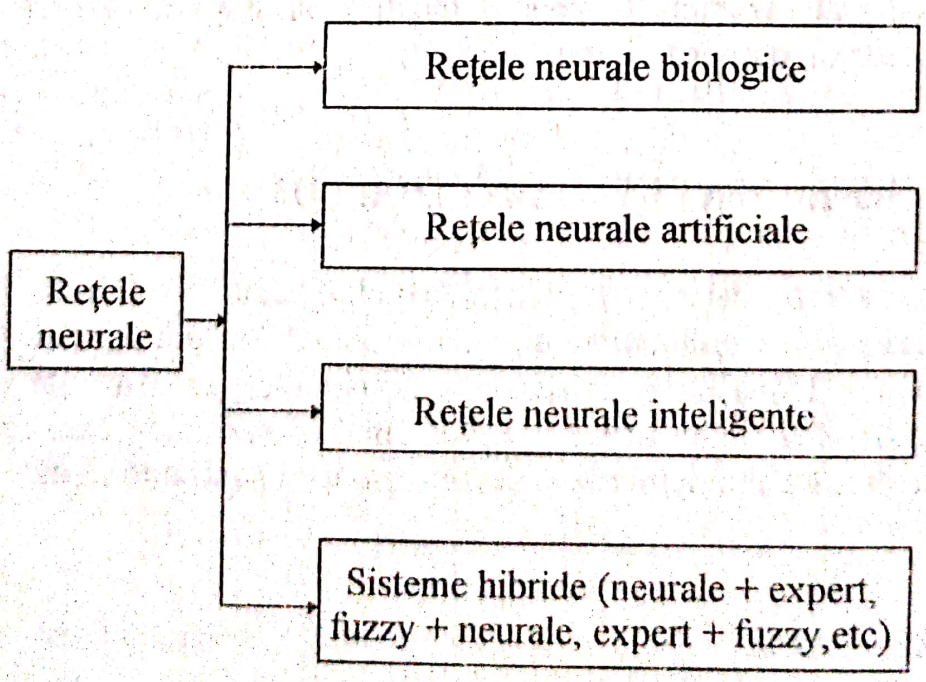


Fig. 3.1 Tipuri de rețele neurale

Scanned with OKEN Scanner

3.1.1 Caracteristicile nodului. Micro – structura neurală

3.1.1.1 Modele de perceptron simplu

Modelul perceptronului simplu (neuron artificial) trebuie să fie, în primul rând, simplu şi să preia principalele caracteristici ale neuronului biologic [B9]: ❶ funcţia de bază a neuronului biologic, de a suma semnalele de intrare şi de a furniza un semnal de ieşire pe baza comparării acestei sume cu un prag prestabilit ("threshold"); ❷ semnalul de ieşire să fie ON sau OFF; ❸ semnalul de ieşire să depindă numai de semnalele de intrare; ❹ intensitatea şi eficienţa sinapselor să poată fi modelate prin introducerea unui factor de multiplicare numit pondere ("weight") pentru fiecare conexiune de intrare a neuronului.

Primul model de perceptron simplu a fost propus în anul 1943 de McCulloch şi Pitts. Ulterior au apărut şi alte modele, cu şi fără reacţie locală la nivelul structurii [C16, D37, H7, T19, ş.a.], sintetizate în tab. 3.1.

Tab. 3.1 Micro – structuri neurale

Tipul micro – structurii neurale	Micro – structura neurală (perceptron simplu)
Fără reacţie	McCulloch – Pitts (perceptron simplu socratic)
	Dahanayake – Upton (perceptron simplu "deştept")
	McCulloch – Pitts modificat
	Fukushima
	Alte modele
Cu reacţie	Lapedes – Farber
	Back – Tsoi
	Frasconi – Gori – Soda
	Frasconi – Gori – Soda generalizat
	DeVries – Principe
	Poddar – Unnikrishnan
	Dahanayake – Upton (perceptron simplu "inteligent")
	Hopfield
	Grossberg
	LRGF ("Locally Recurrent Globally Feedforward")

Scanned with OKEN Scanner

❑ *Modele de perceptron simplu fără reacţie*

Un model general de perceptron simplu fără reacţie este prezentat în fig. 3.2. Dacă modulul "operaţie aritmetică" este compus din succesiunea "sumare + comparare cu prag", se obţine modelul McCulloch şi Pitts (fig. 3.3). Prin particularizări diferite ale aceluiaşi modul, se obţin modelele de perceptron simplu de ordin superior ("high order"), cu funcţii de activare polinomiale, "ridge", sigma – pi, etc.

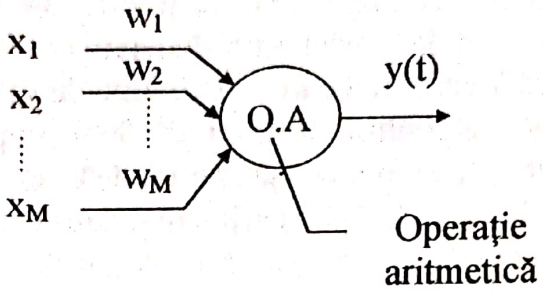


Fig. 3.2 Model general de perceptron simplu fără reacţie

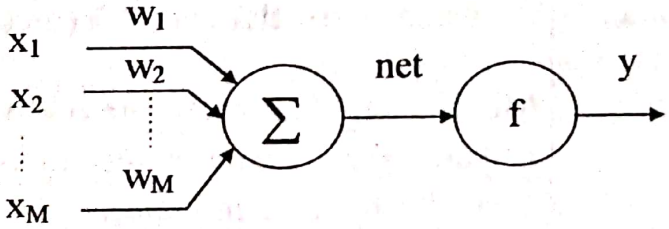


Fig. 3.3 Modelul perceptronului simplu McCulloch – Pitts

S-au notat cu:

$\mathbf{x} = [x_1, x_2, \ldots, x_M]$	vectorul de intrare
$\mathbf{w} = [w_1, w_2, \ldots, w_M]$	vectorul ponderilor
f	funcţia de activare[6] a perceptronului simplu
y	semnalul de ieşire

Ponderile sunt valori constante (la trecerea semnalului în sens intrare – ieşire), ce caracterizează conexiunile de intrare în perceptronul simplu. Valoarea semnalului de ieşire *y* al perceptronului simplu McCulloch – Pitts este dată de relaţia (3.1), unde s-a notat cu *net*, produsul scalar al vectorului transpus al ponderilor **w** şi al vectorului de intrare ***x*** (3.2):

[6] Funcţia de activare se mai numeşte "caracteristică de transfer a perceptronului simplu", "funcţie prag", etc. ("Threshold function", "Squashing function", etc.) [C16]

Scanned with OKEN Scanner

$$y = f(net) = f\left(\sum_{i=1}^{M} w_i\, x_i\right) \tag{3.1}$$

$$net = \mathbf{w}^T \mathbf{x} = \sum_{i=1}^{M} w_i\, x_i \tag{3.2}$$

Dacă se consideră că perceptronul simplu are poziţia *j* într-o structură cu mai mulţi perceptroni simpli, atunci expresiile (3.1), (3.2) primesc un indice suplimentar. Valoarea ieşirii perceptronului simplu este:

$$y_j = f_j(net_j) = f_j\left(\sum_{i=1}^{M} w_{ij}\, x_i\right) \tag{3.3}$$

Funcţionarea modelului McCulloch – Pitts de perceptron simplu constă în realizarea sumării ponderate a semnalelor de intrare şi calculul valorii funcţiei de activare f_j, de argument net_j. Funcţia de activare este funcţia treaptă Heaviside (3.4). In consecinţă, valoarea ieşirii y_j este +1 dacă $net_j > b_j$, respectiv 0 altfel. S-a notat cu b_j valoarea pragului de comparaţie [B9].

$$f_j(net_j) = \begin{cases} 1 & \text{pentru } net_j > b_j \\ 0 & \text{altfel} \end{cases} \tag{3.4}$$

Echivalent comparaţiei, se poate considera funcţia treaptă deplasată dată de relaţia (3.5), din care rezultă că se poate introduce o intrare suplimentară, având ponderea fixă de valoare $-b_j$ şi valoarea intrării $x_0 = 1$, astfel încât valoarea ieşirii perceptronului simplu să se poată scrie conform relaţiei (3.6).

In continuarea acestui paragraf vom renunţa la indicele *j*.

$$y_j + b_j = f_j\left(\sum_{i=1}^{M} w_{ij}\, x_i\right) \Rightarrow y_j = f_j\left(\sum_{i=1}^{M} w_{ij}\, x_i - b_j\right) \tag{3.5}$$

$$y_j = f_j\left(\sum_{i=0}^{M} w_{ij}\, x_i\right) \tag{3.6}$$

Modelul McCulloch – Pitts este numit şi perceptron simplu (neuron artificial) convenţional sau socratic, datorită utilizării funcţiei de activare treaptă. Motivând denumirea, Bandula Dahanayake afirmă [D37]:

Scanned with OKEN Scanner

Lumea este invadata de gândirea socratica. Aceasta se bazeaza pe:

- *dihotomii*
- *polaritati (+ /−, adevarat / fals, tot / nimic)*

Formularea critica a întrebarilor reprezinta baza gândirii socratice.

subliniind apoi faptul că, pentru a se obține structuri neurale performante, este necesară înlocuirea perceptronului simplu socratic cu perceptronul simplu "deștept" ("smart neuron") sau, cu perceptronul simplu "inteligent" ("intelligent neuron").

❑ *Modele de perceptron simplu cu reacție*

Există mai multe metode de includere a reacției ("feedback") la nivelul perceptronului simplu din fig. 3.3: la nivelul activării locale (fig. 3.4), la nivelul sinapsei (fig. 3.5) sau la nivelul ieșirii (fig 3.6).

In cazul perceptronului simplu cu reacție la nivelul activării locale, valorile de intrare pot fi replici întârziate ale valorilor de activare *net*:

$$x_i(t) = net(t - i) \tag{3.7}$$

(unde $1 \le i \le M$ este indicele valorilor de intrare, iar t este timpul; s-a utilizat o notație simplificată pentru conținutul parantezei din dreapta egalității, care este de fapt $t - iT$, cu T este perioada de eșantionare, considerată unitară) sau combinații ale versiunilor întârziate ale valorilor *net* și ale valorilor de intrare întârziate (fig. 3.4) [T19].

In perceptronul simplu cu reacție locală la nivelul sinapsei (fig. 3.5), fiecare funcție sinaptică este modelată printr-o funcție liniară de transfer G_1, ... G_M. Activările locale ale sinapselor se sumează la intrarea elementului neliniar din structură. Se poate observa că, reacția locală la nivelul activării este un caz particular al reacției locale la nivelul sinapsei.

In fig. 3.6, se observă că, valorile de intrare în perceptronul simplu cu reacție la ieșire, sunt valori de ieșire întârziate.

Hecht – Nielsen [H7] a propus un model de perceptron simplu cu reacție locală generică, care include un modul de memorie în cadrul structurii. Pe de altă parte, Tsoi și Back [T19] au arătat că, modelul general de perceptron simplu (fig. 3.7), conține cazurile particulare cu reacție la nivelul sinapsei, precum și pe cele cu reacție la ieșire. Acesta stă la baza rețelelor ce au reacție locală, dar sunt "feedforward" la nivel global (GLRGF = "General Locally Recurrent Globally Feedforward Network").

In consecință, modelele Lapedes – Farber, Back – Tsoi, Frasconi – Gori – Soda, Frasconi – Gori – Soda generalizat, DeVries – Principe, Poddar –

Scanned with OKEN Scanner

Unnikrishnan, Dahanayake – Upton, Hopfield şi Grossberg (anexa 3), care sunt cazuri particulare ale structurilor cu reacţie din fig. 3.4 – 3.6, sunt totodată cazuri limită ale GLRGF.

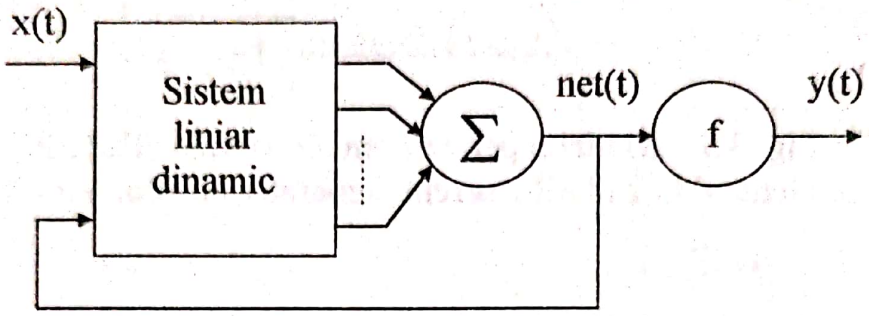


Fig. 3.4 Modelul perceptronului simplu cu reacţie la nivelul activării locale

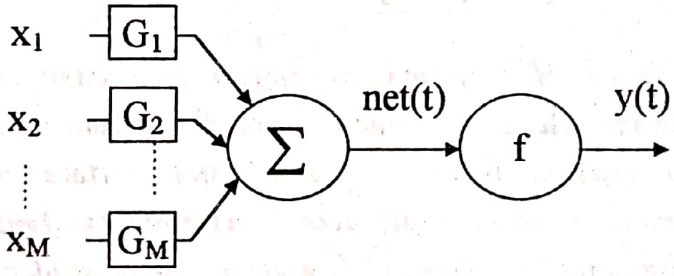


Fig. 3.5 Modelul perceptronului simplu cu reacţie locală la nivelul sinapsei

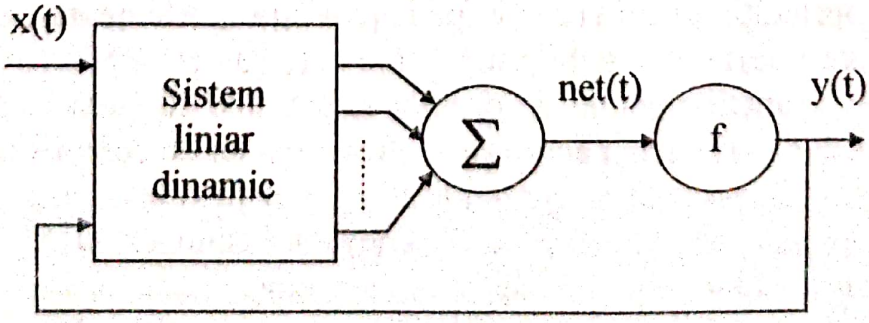


Fig. 3.6 Modelul perceptronului simplu cu reacţie locală la ieşire

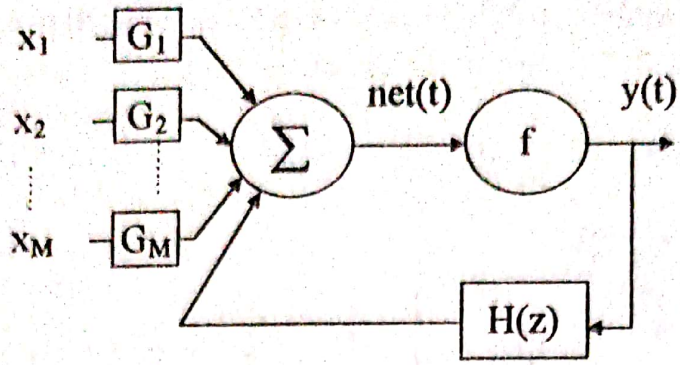


Fig. 3.7 Modelul perceptronului simplu GLRGF
("General Locally Recurrent Generally Feedforward")

3.1.1.2 Funcţii de activare

Modelul de perceptron simplu McCulloch – Pitts este caracterizat de funcţia de activare de tip prag binar (sau bipolar). In consecinţă, valoarea de ieşire a perceptronului simplu poate fi 0 sau +1. Limita impusă modelului de perceptron simplu socratic de funcţia de activare de tip prag este comentată astfel de Bandula Dahanayake [D37]:

> *Modelul cu prag, cu inflexibilitatea sa caracteristica, determina rezultatul sa aiba o valoare extrema sau alta, atunci când ar putea fi mai bine ca acesta sa aiba o valoare între cele doua extreme. Invatarea rapida necesita libertate intelectuala. Neuronii si retelele conventionale nu au exact aceasta libertate esentiala, care este un factor indispensabil învatarii. Gânditi-va, de exemplu, de ce studentilor la universitate si profesorilor li se permite sa aiba libertate intelectuala. De aceea, prima noastra sarcina ar trebui sa fie aceea de a elibera neuronii, de a le da libertate intelectuala.*

Este posibilă utilizarea altor funcţii de activare neliniare, monotone sau nemonotone (anexa 4), a căror utilitate devine evidentă odată cu precizarea legii de învăţare a structurii neurale. Pe baza teoriei aproximării funcţiilor pe de o parte, şi a proprietăţii de aproximare pe care o au reţelele neurale (par. 3.2.2.1), au fost utilizate o serie de funcţii printre care polinoamele algebrice, polinoamele trigonometrice, funcţiile spline de ordinul *s*, funcţiile cu bază radială, funcţiile "ridge", etc. [W27]. Eficienţa acestora în structuri neurale compuse din mai mulţi perceptroni simpli, precum şi condiţiile în care sunt echivalente cu funcţia de activare sigmoidală sunt discutate de DasGupta şi Schnitger [D15].

Clasificarea funcţiilor de activare se poate face conform criteriilor: liniaritate / neliniaritate, monotonicitate / nemonotonicitate, domeniul de valori (continuu / discret, binar / bipolar / real). Alte clasificări consideră două categorii

Scanned with OKEN Scanner

de perceptroni simpli: cei definiţi de relaţia (3.6) şi perceptroni simpli flexibili, din cea de-a doua categorie făcând parte modelele cu funcţii de activare "spline", "projection pursuit", "ridge polynomial" [K33]. Vom considera clasificarea prezentată în tab. 3.2, unde am indicat prin *Mn* monotonicitatea funcţiei, respectiv *Nm* nemonotonicitatea acesteia.

Tab. 3.2 Clasificarea funcţiilor de activare

Tipul funcţiei			Caracteristici
Funcţie deterministă	Prag		Unipolar	Mn
			Bipolar	Mn
			Cu zonă moartă	Nm
			Cu histerezis	Nm
	Liniară	Rampă	Unipolară	Mn
			Bipolară	Mn
		Modul		Nm
		Triunghiulară		Nm
		Trapezoidală		Nm
	Neliniară	Sigmoidală M		Mn
		Sigmoidală N		Nm
		Polinomială trigonometrică		Mn / Nm
		RBF	Gausiană	Nm
			Pălărie mexicană	Nm
			Alte funcţii	Nm
		"Ridge"		Mn / Nm
		De ordin superior (HONN)	Sigma – pi	Mn / Nm
			Spline	Mn / Nm
			Polinomială algebrică	Mn / Nm
			"Projection pursuit"	Nm
		Alte funcţii		Mn / Nm
Funcţie stohastică				Mn

Dintre funcţiile de activare neliniare, cea sigmoidală monotonă unipolară, definită de relaţia (3.8) şi cea bipolară de (3.9), sunt deseori utilizate. Alte alegeri posibile de funcţii de activare sigmoidale sunt (3.10) sau (3.11) [F18]. S-a notat cu *s* panta funcţiei.

Scanned with OKEN Scanner

$$f(net) = \frac{1}{1 + \exp(-s\ net)} \tag{3.8}$$

$$f(net) = \frac{1 - \exp(-s\ net)}{1 + \exp(-s\ net)} \tag{3.9}$$

$$f(net) = \tanh(s\ net) = \frac{1 - \exp(-2\,s\,net)}{1 + \exp(-2\,s\,net)} \tag{3.10}$$

$$f(net) = \tanh(s\ net) + 0.1\ net \tag{3.11}$$

3.1.2 Topologia reţelei. Mezo–structura şi macro–structura neurală

Nivelele ierarhice superioare micro – structurii din punct de vedere arhitectural, sunt mezo– şi respectiv macro – structura, fiecare cu topologii specifice. Trebuie precizat faptul că, deşi există excepţii, în general în domeniul ştiinţelor conexioniste, care include şi RNA, cuvântul "arhitectură" se referă la descrierea matematică formală a reţelei neurale artificiale, şi nu la implementarea specifică a acesteia. De aceea, în continuare vom avea în vedere această semnificaţie, realizarea practică hardware şi / sau software fiind abordată de alte lucrări [B35, F32, H44, J12, L41, M52, M70, O7, P25, ş.a.].

3.1.2.1 Mezo – structura neurală

Interconectarea perceptronilor simpli conform unei topologii alese, conduce la structura (reţeaua), care constituie nivelul de mezo – structură neurală. Este esenţial faptul că, sistemul neural rezultat nu este o sumă a componentelor sale. Altfel spus, proprietăţile sale globale nu se obţin sumând calităţile individuale ale perceptronilor simpli. Afirmaţia este mai generală, nu se aplică numai în cazul reţelelor neurale artificiale.

La nivel de mezo – structură, trebuie luat în considerare modul de interconectare a perceptronilor simpli (topologia reţelei), cu precizarea numărului acestora, numărului straturilor (dacă este cazul), gradului de conectivitate (definit ca numărul total al conexiunilor structurii), tipului conexiunilor, etc. In fig. 3.8 este prezentată o clasificare posibilă a mezo – structurilor neurale.

Mezo – structurile neurale pot fi definite în general, pornind de la precizările lui Hecht -- Nielsen.

Scanned with OKEN Scanner

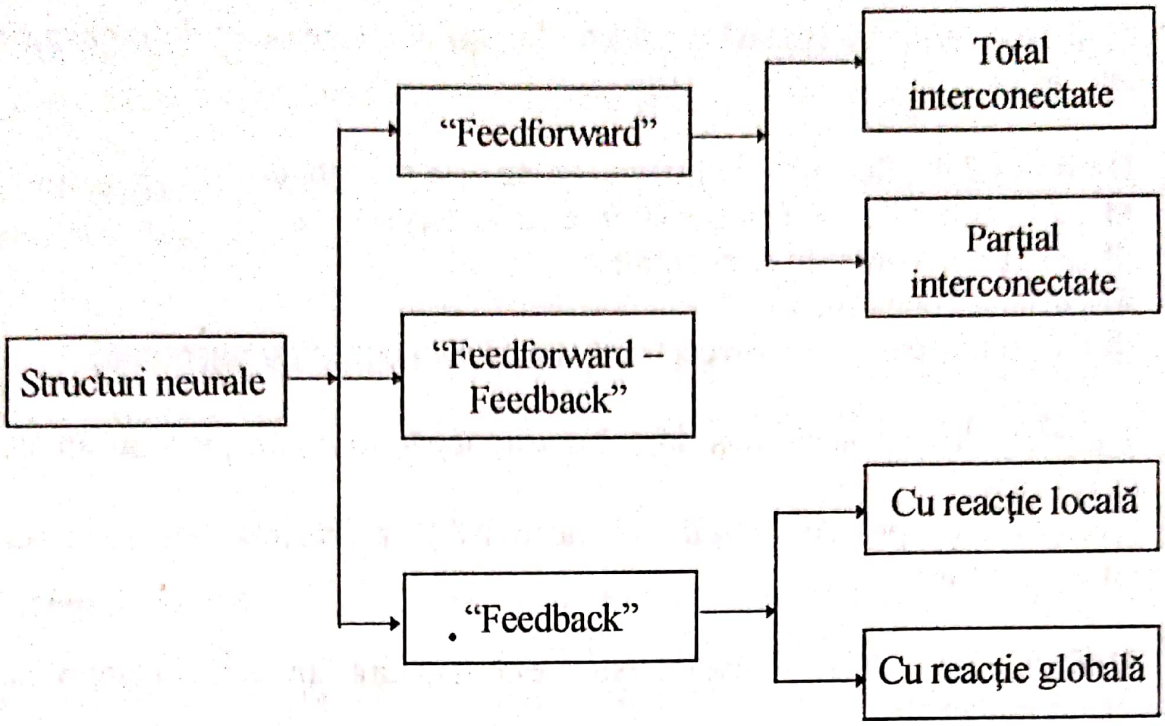


Fig. 3.8 Clasificarea mezo–structurilor neurale

Definiţia 3.1 Mezo – structura neurală este structura paralelă şi distribuită de prelucrare a informaţiei, constituită prin conectarea micro – structurilor neurale perceptron simplu, cu următoarele precizări şi restricţii:

- Componentele micro – structurale se numesc elemente de prelucrare (sau noduri, neuroni, neuroni formali, neurode, perceptroni simpli).
- Elementele de prelucrare sunt legate prin intermediul conexiunilor.
- Fiecare element de prelucrare poate primi orice număr de conexiuni (numite conexiuni de intrare).
- Fiecare element de prelucrare poate avea oricâte conexiuni de ieşire, dar valoarea semnalului în fiecare dintre acestea este aceeaşi. In consecinţă, fiecare element de prelucrare are o singură conexiune de ieşire.
- Elementele de prelucrare pot avea memorie locală.
- Fiecare element de prelucrare este caracterizat de o funcţie de transfer, numită funcţie de activare, care poate opera continuu sau nu.

Definiţia 3.2 Topologia reţelei este definită de geometria sa şi de schema de interconectare [F37].

Definiţia 3.3 Geometria mezo – structurii neurale este perechea (S, g), constând din setul S compus din Q puncte, cunoscut sub numele de grilă (nu neapărat dreptunghiulară) în spaţiul euclidian M – dimensional, $M \leq Q$,

Scanned with OKEN Scanner

împreună cu o proiecţie (notată cu g) unu la unu, a elementelor de prelucrare în punctele grilei.

Definiţia 3.4 Schema de interconectare este definită de:
❶ Tipul conexiunilor (intrastrat, interstrat, suprastrat)
❷ Simetria / asimetria conexiunilor
❸ Ordinul conexiunilor
❹ Conectivitatea (care perceptroni simpli sunt conectaţi împreună)

Definiţia 3.5 Conexiunea intrastrat este legătura dintre perceptroni simpli din acelaşi strat.

Conexiunile proprii ("self connections") reprezintă o subclasă a conexiunilor intrastrat.

Definiţia 3.6 Conexiunea interstrat este legătura dintre perceptroni simpli din straturi adiacente[7].

Definiţia 3.7 Conexiunea proprie este legătura dintre origine şi punctul terminus în acelaşi perceptron simplu. Conexiunea proprie este cea care realizează reacţia la nivelul perceptronului simplu [F37].

Definiţia 3.8 Conexiunea suprastrat este legătura dintre perceptronii simpli care nu fac parte nici din straturi adiacente, nici din acelaşi strat.

Definiţia 3.9 O mezo – structură neurală plenară este RNA în care există toate tipurile de conexiuni: intrastrat, interstrat, suprastrat. Este o reţea total interconectată în adevăratul sens al cuvântului.

Definiţia 3.10 Conexiunea simetrică sau bidirecţională este legătura caracterizată de aceeaşi valoare a ponderii, când este utilizată în ambele direcţii de transmitere a semnalului.

Definiţia 3.11 Conexiunea asimetrică sau unidirecţională este legătura ce poate fi utilizată într-o singură direcţie de transmitere a semnalului.

Definiţia 3.12 Mezo – structura simetrică este RNA care conţine numai conexiuni simetrice. Matricea sa de ponderi este simetrică.

Definiţia 3.13 Mezo – structura asimetrică este RNA care conţine una sau mai multe conexiuni asimetrice [F37].

[7] Straturi adiacente sunt cele pentru care indexul stratului diferă printr-o unitate.

Scanned with OKEN Scanner

Definiţia 3.14 Mezo – structura neurală de ordinul s este RNA compusă din perceptroni simpli, dintre care cel puţin unul are ordinul egal cu s.

Definiţia 3.15 Mezo – structura de tip "feedforward" este reţeaua topologic reprezentabilă ca un graf orientat.

Această definiţie este prea generală pentru a caracteriza corespunzător mezo – structura "feedforward". De aceea, prin reformularea definiţiilor 3.1 şi 3.3, se obţin:

Definiţia 3.16 Mezo – structura neurală "feedforward" este structura paralelă şi distribuită de prelucrare a informaţiei, sub forma unui graf orientat. Este constituită prin conectarea micro – structurilor neurale perceptron simplu, cu următoarele precizări şi restricţii:

- Nodurile grafului se numesc elemente de prelucrare (componente micro–structurale, noduri, neuroni, neuroni formali, neurode, perceptroni simpli).
- Ramurile grafului se numesc conexiuni.
- Fiecare element de prelucrare poate primi orice număr de conexiuni (numite conexiuni de intrare).
- Fiecare element de prelucrare poate avea oricâte conexiuni de ieşire, dar valoarea semnalului în fiecare dintre acestea este aceeaşi. In consecinţă, fiecare element de prelucrare are o singură conexiune de ieşire.
- Elementele de prelucrare pot avea memorie locală.
- Fiecare element de prelucrare este caracterizat de o funcţie de transfer, numită funcţie de activare, care poate opera continuu sau discret.

In plus, structura neurală are elementele de prelucrare divizate în subseturi disjuncte, numite straturi, în care de obicei funcţia de activare este aceeaşi.

Definiţia 3.17 Geometria unui strat al mezo – structurii neurale este perechea (S, g), constând din setul S compus din Q puncte, cunoscut sub numele de grilă (nu neapărat dreptunghiulară) în spaţiul euclidian M – dimensional, $M \leq Q$, împreună cu o proiecţie (g) unu la unu, a elementelor de prelucrare din strat în punctele grilei. Elementele de prelucrare ale stratului sunt numerotate de la 1 la M.

Definiţia 3.18 Geometria mezo – structurii neurale artificiale este o colecţie a geometriilor tuturor straturilor sale [H7].

Scanned with OKEN Scanner

❒ *Mezo – structura neurală "feedforward"*

Definiţiile 3.2, 3.4 – 3.16, caracterizează mezo – structura "feedforward", reprezentată grafic în fig. 3.9.

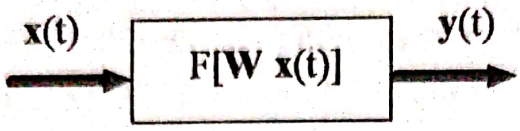


Fig. 3.9 Mezo – structura neurală "feedforward"

S-au folosit notaţiile:

$\mathbf{x} = [x_1, x_2, ..., x_M]$	vector de intrare
$\mathbf{y} = [y_1, y_2, ..., y_N]$	vector de ieşire
w_{ij}	ponderea conexiunii dintre perceptronii simpli *i* şi *j*
$\mathbf{w}_j$	vectorul ponderilor conexiunilor dintre neuronii de intrare *i* şi neuronul de ieşire *j*
$\mathbf{W}$	matricea de ponderi ce are drept coloane vectorii $\mathbf{w}_j$
F	funcţie neliniară

Funcţia neliniară *F* realizează transformarea (3.12) pentru sisteme continue, respectiv (3.13) pentru sistemele discrete [Z2]. S-a notat cu *k* indexul discret al timpului.

$$\mathbf{y}(t) = F[\mathbf{W}\ \mathbf{x}(t)] \tag{3.12}$$

$$\mathbf{y}(k) = F[\mathbf{W}\ \mathbf{x}(k)] \tag{3.13}$$

O clasificare a mezo – structurilor "feedforward" este prezentată în tab. 3.3. Structurile "feedforward" la nivel local şi global sunt constituite din perceptroni simpli fără reacţie (par. 3.1), de tip McCulloch – Pitts, McCulloch – Pitts modificat sau alte modele (sigma – pi, pi – sigma, etc.). Din această clasă fac parte combinatorul liniar adaptiv ALC ("Adaptive Linear Combiner"), perceptronul, Adaline, Madaline, perceptronul multistrat, structurile RBF, maşina Boltzmann, maşina Cauchy, etc.

In categoria mezo – structurilor neurale "feedforward" cu reacţie locală se pot include modelele care sunt formate prin interconectarea perceptronilor simpli cu reacţie. Reţelele rezultate sunt "feedforward" la nivel global, dar recurente la nivelul elementului de prelucrare şi sunt numite LRGF ("Locally Recurrent

Scanned with OKEN Scanner

Globally Feedforward"). Modelele Lapedes – Farber, Back – Tsoi, Frasconi – Gori – Soda, Frasconi – Gori – Soda generalizat, DeVries – Principe, Poddar – Unnikrishnan, Dahanayake – Upton, precum şi modelul general LRGF ("Locally Recurrent Globally Feedforward") fac parte din această categorie.

Tab. 3.3 Clasificarea mezo – structurilor "feedforward"

Mezo – structuri "feedforward"		
"Feedforward" la nivel local şi global	Formate prin interconectarea perceptronilor simpli McCulloch– Pitts	Perceptron
		Combinator liniar adaptiv
		Adaline
		Madaline
	Formate prin interconectarea perceptronilor simpli cu alte funcţii de activare	Perceptron "deştept"
		Perceptron multistrat
		Reţea polinomială trigonometrică
		Reţea RBF
		Maşina Boltzmann
		Maşina Cauchy
	Formate prin interconectarea altor perceptroni simpli fără reacţie	Reţea de ordin superior sigma – pi
		Reţea de ordin superior "ridge"
		Reţea de ordin superior spline
		Reţea polinomială algebrică
		Cognitron, neocognitron
"Feedforward" la nivel global, cu reacţie locală	Lapedes – Farber	
	Back – Tsoi	
	Frasconi – Gori – Soda	
	Frasconi – Gori – Soda generalizat	
	DeVries – Principe	
	Poddar – Unnikrishnan	
	Dahanayake – Upton	
	LRGF ("Locally Recurrent Globally Feedforward")	

❒ ***Mezo – structura neurală "feedback"***

Mezo – structurile recurente sunt caracterizate de definiţiile generale 3.1 – 3.4, cărora li se adaugă definiţiile 3.10 – 3.13. Reprezentarea grafică a reţelei neurale recurente este realizată în fig. 3.10, unde semnificaţia notaţiilor este aceeaşi ca în fig. 3.9. Funcţia neliniară F realizează transformarea (3.14) pentru

Scanned with OKEN Scanner

sistemele continue, respectiv (3.16) pentru cele discrete, şi condiţia iniţială (3.15). S-a notat cu *k* indexul discret al timpului şi cu *Δ* întârzierea.

$$y(t + \Delta) = F\,[W\ y(t)] \tag{3.14}$$

$$y(0) = x(0) \tag{3.15}$$

$$y(k + 1) = F\,[W\ y(k)] \tag{3.16}$$

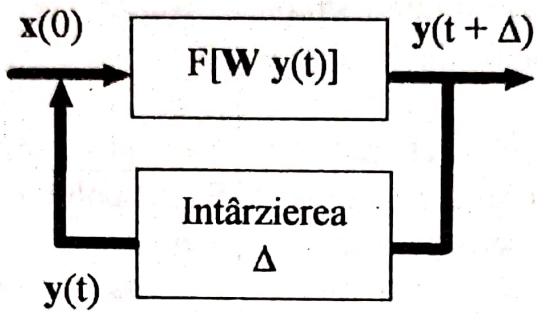


Fig. 3.10 Mezo – structura neurală "feedback"

Modelele recurente se numesc "parţial recurente" dacă pot fi reprezentate ca o succesiune de straturi (strat de intrare, strat ascuns, strat de ieşire) şi "propriu – zis recurente" altfel. La rândul lor, mezo – structurile parţial recurente (numite şi "reţele cu noduri de context" [H19]) pot avea reacţie locală sau globală. Din prima categorie fac parte cele în care reacţia există la nivelul perceptronului simplu, dar structura globală este "feedforward". Acestea au fost amintite anterior. Modelele cu reacţie globală prezintă conexiuni inverse la nivelul mezo – structurii, iar din această clasă fac parte modelele Williams – Zipser, Jordan, Elman, Jordan – Elman, Stornetta, Mozer, ş.a.

❒ ***Relaţia dintre mezo – structura "feedforward" şi cea "feedback"***

Relaţia dintre mezo – structura neurală "feedforward" şi cea "feedback" este bidirecţională. Pe de o parte, din mezo – structura recurentă generală (fig. 3.10) se poate obţine mezo – structura generală "feedforward", prin anularea tuturor conexiunilor inverse în reţea şi la nivelul perceptronilor simpli. Din acest punct de vedere, se poate considera că modelul "feedforward" este o particularizare a celui cu reacţie.

Pe de altă parte, s-a arătat [N15] că, orice structură recurentă poate fi reprezentată sub formă canonică, constând dintr-o reţea (statică) de tip

Scanned with OKEN Scanner

"feedforward", cu caracteristicile: ❶ intrările sale sunt de două tipuri: intrări propriu–zise în rețea și intrări ce conțin valorile variabilelor de stare, cele din urmă fiind întârziate cu un pas; ❷ ieșirile sale sunt de două tipuri: ieșiri propriu–zise și ieșiri care sunt variabile de stare. Din acest punct de vedere, se poate considera că modelul "feedback" reprezintă o extensie a modelelor "feedforward". In favoarea acestei afirmații formulează argumente și Hecht – Nielsen, care consideră că, orice structură neurală cu reacție poate fi privită ca o rețea "feedforward", în care fiecare strat conține câte un singur perceptron simplu. Atunci, definirea generală a mezo – structurii neurale artificiale (atât "feedforward" cât și "feedback"), se realizează pe baza definițiilor 3.2, 3.16 – 3.18, cărora li se adaugă 3.5 – 3.14.

3.1.2.2 Macro – structura neurală

La nivel de macro – structură, arhitecturile neurale se compun din mezo – structuri, scopul utilizării fiind rezolvarea unor probleme de mari dimensiuni. Sistemele de rețele neurale, cum mai sunt numite, pot avea: cuplare/interconectare puternică sau cuplare/interconectare slabă [P201, R3, H7]. Macro – structura neurală cu interconectare puternică poate fi considerată o unică rețea neurală de dimensiuni mari, realizată prin fuzionarea a două sau mai multe mezo – structuri. In macro – structurile neurale cu interconectare slabă, mezo – structurile neurale componente își păstrează individualitatea [C40]. Acestea pot fi dispuse în paralel sau ierarhic. Alteori, sistemele cu cuplare slabă pot utiliza o mezo – structură ca preprocesor al altei mezo – structuri neurale. Indiferent de tipul cuplării existente la nivelul macro – structurii, proiectarea sistemelor de rețele neurale nu se limitează la alegerea mezo – structurilor componente, ci presupune crearea întregii arhitecturi pentru rezolvarea unei probleme complexe.

3.1.3 Modelul datelor și criteriul de cost

Caracterizarea unei RNA la nivel de micro–, mezo– și macro – structură, presupune și precizarea problemei date, rezolvabilă cu ajutorul modelului neural obținut. Din punct de vedere al acestuia, aplicația este caracterizată prin datele disponibile, care determină alegerea unui criteriu (funcție) cost, în conformitate cu care rețeaua neurală învață și este evaluată (par. 3.1.4). Funcția cost nu caracterizează direct modelul neural propriu – zis, ci reprezintă o anumită ipoteză asupra distribuției datelor de intrare. Pentru RNA de tip "feedforward", funcția se notează de obicei cu $C(\mathbf{w})$, este definită pe spațiul ponderilor (parametrii rețelei) și se numește criteriu de cost. Se urmărește determinarea

vectorului de ponderi $\mathbf{w} \in \Re^{\Pi}$ pentru care $C(\mathbf{w})$ este minimizată (relaţia (3.17)). Vectorul ponderilor este Π – dimensional şi poate conţine şi valorile pragurilor, sau alţi parametri ai reţelei. Funcţia $C(\mathbf{w})$ trebuie să satisfacă teorema 3.1 [C16].

Minimizează funcţia scalară de cost $C(\mathbf{w})$ cu constrângerile $\mathbf{w} \in \Re^{\Pi}$	(3.17)

Teorema 3.1 Fie matricea $\nabla^2 C(\mathbf{w})$, nesingulară în punctul $\mathbf{w}^*$. Dacă matricea $\nabla C(\mathbf{w}^*) = 0$ şi dacă matricea hessiană $\nabla^2 C(\mathbf{w}^*)$ este simetrică şi pozitiv definită, atunci $C(\mathbf{w}^*) < C(\mathbf{w})$ pentru fiecare vector $\mathbf{w}$, care satisface relaţia $0 < \| \mathbf{w} - \mathbf{w}^* \| < \varepsilon$, cu $\varepsilon > 0$.

3.1.4 Legea de învăţare

3.1.4.1 Definiţiile învăţării

In sistemele naturale (biologice), învăţarea a fost definită mai ales ca o modificare de comportament. Cercetătorii se referă deseori la învăţare, ca la o "condiţionare", descriind tipul "clasic", evaluat de fiziologul rus Pavlov şi tipul bazat pe legea efectului Thorndike (numit şi condiţionare instrumentală) [M17, Z2]. Incercând să aprecieze învăţarea într-un context general, Maren, Harston şi Pap consideră că, "sistemele inteligente sunt definite prin ceea ce fac. Invăţarea este un proces inteligent" [M17]. Pe de altă parte, calitatea de "inteligent" a sistemului este definită de Dahanayake drept "capacitatea de a fi conştient şi de a utiliza inovaţiile trecute şi prezente, pentru decizii viitoare într-un mod lipsit de emotivitate" [D37].

In sistemele artificiale, pot fi utilizate definiţiile 3.19 – 3.21.

Definiţia 3.19 *Sistemele inteligente învata sau se adapteaza. "Invatarea" sau "adaptarea" sunt daruri lingvistice din antichitate. Acestea înseamna pur si simplu "modificarea parametrilor" (Bart Kosko).*

Definiţia 3.20 *Invatarea poate fi definita ca o modificare a comportamentului datorata procesului de antrenare. Performanta se evalueaza înainte si dupa procesul de învatare. Diferenta arata cât de mult s-a învatat (Maren, Harston si Pap).*

Scanned with OKEN Scanner

Definiţia 3.21 *Invatarea reprezinta orice modificare mai mult sau mai putin durabila în raspunsul sistemului la prezentarea succesiva a aceluiasi stimul (Zornetzer, Davis si Lau).*

Definiţia 3.21 este prea generală pentru a putea fi utilizată. In continuare vom considera învăţarea un proces de "adaptare" şi modificare a parametrilor modelului, conform definiţiilor 3.19 şi 3.20, valabile atât pentru sisteme biologice cât şi pentru cele artificiale.

Scopul modificării parametrilor modelului neural prin învăţare este minimizarea funcţiei cost (obiectiv, energie, etc.) pentru întreg setul datelor de învăţare. Rezultatul învăţării reţelei neurale este setul de parametri (de obicei ponderi).

Reprezentarea rezultatelor învăţării în reţelele neurale artificiale se referă de obicei la ponderile reţelei. Reprezentarea se poate realiza prin valorile numerice corespunzătoare, sau grafic, prin diagrame care vizualizează ponderile conexiunilor structurii.

3.1.4.2 Legi de învăţare

Imbunătăţirea învăţării în reţelele neurale se poate realiza prin alegerea corespunzătoare a perceptronilor simpli, astfel încât reţeaua totală să înveţe repede şi eficient, şi prin realizarea unor algoritmi de învăţare rapizi. Aceştia se bazează de obicei pe modificarea locală a ponderilor, la nivelul perceptronului simplu, conform relaţiilor matematice numite "legi de învăţare". Clasificarea modelelor neurale din punct de vedere al învăţării este prezentată în tab. 3.4.

Legea de învăţare este supervizată (fig. 3.11) atunci când se prezintă reţelei un set de date de intrare $\{x(\xi)\}$, împreună cu setul corect de semnale de ieşire (dorite) $\{d(\xi)\}$, $1 \le \xi \le P$ pentru acele intrări. Setul de date de învăţare este compus dintr-un număr total de P perechi de învăţare (x, d). Se urmăreşte minimizarea distanţei dintre valorile de ieşire obţinute şi cele dorite, pe baza criteriului de cost ales. Se mai numeşte "învăţare cu profesor", sau "învăţare cu instruire" şi este utilă dacă se pot preciza valorile de ieşire dorite, pentru fiecare pattern de intrare [B14].

Legea de învăţare semisupervizată ("reinforcement learning") este similară învăţării supervizate, cu excepţia faptului că, în locul valorilor de ieşire dorite d, reţeaua primeşte doar o "notă" care precizează cât de bine a învăţat patternurile precedente. Se mai numeşte "învăţare cu critic" [H19]. In fig. (3.12) s-a notat cu r semnalul de penalizare / premiere ("reinforcement").

Legea de învăţare este nesupervizată (fig. 3.13) când se prezintă reţelei numai setul de date de intrare, fără a cunoaşte datele corecte de ieşire corespunzătoare intrărilor date.

Scanned with OKEN Scanner

Tab. 3.4 Clasificarea modelelor neurale din punct de vedere al învăţării

Modele neurale	Legea de modificare a ponderilor
Ponderi fixe	Nu se modifică
Ponderi variabile	Supervizată
	Semisupervizată ("reinforcement learning")
	Nesupervizată

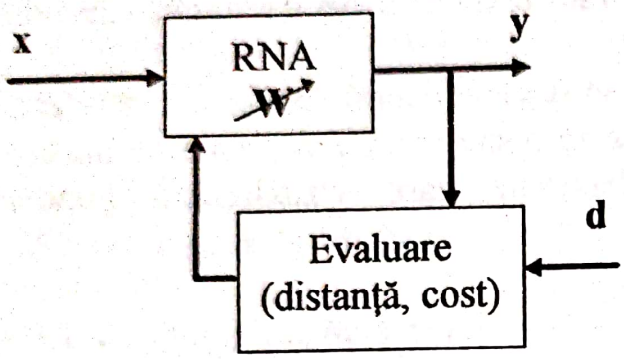


Fig. 3.11 Invăţare supervizată

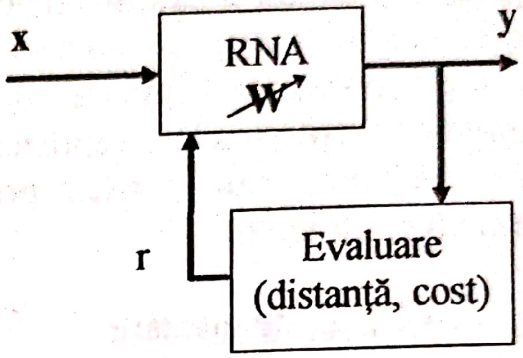


Fig. 3.12 Invăţare semisupervizată ("reinforcement learning")

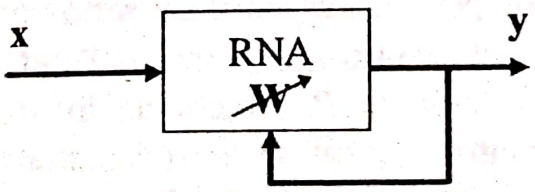


Fig. 3.13 Invăţare nesupervizată

Avantajul legilor de acest tip constă în faptul că nu mai este necesară specificarea explicită a valorilor ieşirilor dorite.

In tab. 3.5 am realizat o clasificare a legilor de învăţare. O parte dintre acestea pot fi obţinute prin particularizarea legii generale de învăţare a modelului propus de Amari în anul 1990 şi sunt menţionate de Zurada [Z2], ş.a.

Scanned with OKEN Scanner

Tab. 3.5 Clasificarea legilor de învăţare

Tipul legii		Legea de învăţare
Supervizată	Bazate pe corectarea erorii	Perceptron
		Delta
		Widrow – Hoff
	Alte legi supervizate	Corelaţie
		Outstar
Semisupervizată		
Nesupervizată	Hebb Anti – Hebb	
	Extrag componente principale	Oja
		Sanger
	Care realizează localizarea	Winner – Take – All
		Rumelhart – Zipser
	Maximizează informaţia	
	Alte legi nesupervizate	Grossberg

3.1.5 Condiţii de stabilitate

Studiul modelelor neurale artificiale, a căror funcţionare este dependentă de timp, trebuie realizat şi din punct de vedere al stabilităţii [N7, P9, ş.a.]. Includerea informaţiei temporale în modelul neural se poate face prin "extensie spaţială" [P22, K27, ş.a.], prin utilizarea modelelor recurente [H11, T20, ş.a.], sau a variantelor derivate din acestea.

Importanţa noţiunii de stabilitate pentru sistemele fizice în general şi pentru sistemele dinamice neurale este susţinută de clasificarea realizată de Benjamin Kao, care le împarte în sisteme stabile şi sisteme inutile. Stabilitatea (sau instabilitatea) sistemului este o caracteristică globală a acestuia, exprimată de criterii precum cel de "intrare – ieşire mărginită" (BIBO, "Bounded Input – Bounded Output") [L200, H11, H21] sau Liapunov. Primul se referă la condiţia impusă ca ieşirea sistemului să nu crească nelimitat, ca rezultat al valorilor de intrare sau al perturbărilor nedorite. Este un criteriu care se aplică sistemelor dinamice liniare, dar practic inutil în cazul sistemelor neurale. Acestea sunt, prin definiţie, stabile în sens BIBO, datorită limitării introduse de funcţia de activare neliniară a perceptronului simplu.

Pentru reţelele neurale artificiale, este utilă abordarea stabilităţii din perspectiva propusă în anul 1892 de Liapunov, care este aplicabilă atât sistemelor liniare, cât şi neliniare, invariante sau nu în timp.

Studiul sistemelor neurale dinamice se poate realiza în mod determinist, caz în care evoluţia modelului în timp este descrisă de un set de ecuaţii diferenţiale

Scanned with OKEN Scanner

neliniare sau din punctul de vedere al neurodinamicii statistice, situaţie în care evoluţia este exprimată de ecuaţii diferenţiale neliniare stohastice. In primul caz, caracterizarea sistemelor dinamice se face pe baza definirii spaţiului stărilor, spaţiului fazelor [C16, S12]. Se formulează condiţia Lipschitz [D4, H41] şi se definesc stările de echilibru. Stabilitatea sistemului neliniar poate fi determinată prin aflarea tuturor soluţiilor posibile ale ecuaţiei în spaţiul stărilor sistemului (variantă dificilă, deseori imposibilă) sau, cu ajutorul teoremelor Liapunov (variantă bazată pe teoremele de stabilitate şi de stabilitate asimptotică ale sistemului autonom dinamic neliniar). Prima teoremă afirmă că, o stare de echilibru dată este stabilă dacă, într-o mică vecinătate a acesteia există o funcţie pozitiv definită dependentă de vectorul de stare, astfel încât derivata funcţiei în raport cu timpul să fie negativ semidefinită în acea regiune [H11, M66]. Cea de-a doua teoremă Liapunov se formulează similar, starea de echilibru fiind asimptotic stabilă dacă există o funcţie pozitiv definită dependentă de vectorul de stare, astfel încât derivata funcţiei să fie negativ definită în acea regiune [K7].

3.1.6 Clasificarea modelelor neurale artificiale

Pe baza caracteristicilor modelelor neurale artificiale prezentate în par. 3.1.1 – 3.1.5, se poate realiza o clasificare a acestora. Criteriile de clasificare a RNA pot fi: tipul modelului neural (fig. 3.1), structura neurală artificială (fig. 3.8, tab. 3.1 – 3.3), legea de învăţare / algoritm (tab. 3.4 –3.5), domeniul de aplicaţie şi tipul acesteia, tipul implementării. Acestora li se pot adăuga funcţia cost şi stabilitatea.

3.2 STRUCTURI NEURALE "FEEDFORWARD"

Structurile neurale artificiale "feedforward" definite în par. 3.1.2 constituie subiectul studiului prezentei lucrări. In continuare vom prezenta aspecte legate de procesul de învăţare, algoritmi de învăţare, generalizare, precum posibilităţi de îmbunătăţire a performanţelor prin reducerea complexităţii modelului "feedforward". Deşi învăţarea reţelelor neurale "feedforward" poate fi atât supervizată cât şi nesupervizată, ne vom referi mai ales la prima dintre acestea.

3.2.1 Notaţii

Notaţiile utilizate în continuare pentru caracterizarea structurii neurale "feedforward", compusă dintr-un strat de intrare, unul sau mai multe straturi ascunse şi un strat de ieşire, sunt:

Scanned with OKEN Scanner

M	numărul perceptronilor simpli din stratul de intrare
H	numărul perceptronilor simpli din stratul ascuns (dacă există mai multe straturi ascunse, atunci numărul elementelor fiecăruia se notează cu H_1, H_2, etc.)
N	numărul perceptronilor simpli din stratul de ieşire
P	numărul patternurilor din setul de învăţare
T	numărul patternurilor din setul de test
ξ	indexul patternului
$\mathbf{x}$	vectorul de intrare; $\mathbf{x} = [x_1, x_2, ..., x_M]$
$\mathbf{x}(\xi)$	vectorul de intrare, pentru patternul ξ; $\mathbf{x}(\xi) = [x_1(\xi), x_2(\xi),..., x_M(\xi)]$
$\mathbf{X}$	matricea vectorilor de intrare; $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ... , \mathbf{x}_P]^T$
$\mathbf{y}$	vectorul de ieşire obţinut; $\mathbf{y} = [y_1, y_2, ..., y_N]$
$\mathbf{y}(\xi)$	vectorul de ieşire obţinut, pentru patternul de intrare ξ; $\mathbf{y}(\xi) = [y_1(\xi), y_2(\xi),...., y_N(\xi)]$
$\mathbf{Y}$	matricea valorilor de ieşire ale celor N perceptroni simpli de ieşire, pentru toate patternurile de învăţare; $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, ... , \mathbf{y}_P]^T$
$\mathbf{d}$	vectorul de ieşire dorit; $\mathbf{d} = [d_1, d_2, ..., d_N]$
$\mathbf{d}(\xi)$	vectorul de ieşire dorit, pentru patternul de intrare ξ; $\mathbf{d}(\xi) = [d_1(\xi), d_2(\xi), ..., d_N(\xi)]$
$\mathbf{D}$	matricea valorilor de ieşire dorite pentru cei N perceptroni simpli de ieşire; $\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, ... , \mathbf{d}_P]^T$
$\mathbf{z}$	vectorul de ieşire al perceptronilor simpli din stratul ascuns; $\mathbf{z} = [z_1, z_2,..., z_H]$
$\mathbf{z}(\xi)$	vectorul de ieşire al perceptronilor simpli din stratul ascuns, pentru patternul de intrare ξ; $\mathbf{z}(\xi) = [z_1(\xi), z_2(\xi), ..., z_H(\xi)]$
$\mathbf{Z}$	matricea valorilor de ieşire a celor H perceptroni simpli ascunşi, pentru toate patternurile de învăţare; $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, ... , \mathbf{z}_P]^T$
w_{ih}	ponderea conexiunii dintre nodul i din stratul de intrare şi nodul h din stratul ascuns
v_{hj}	ponderea conexiunii dintre nodul h din stratul ascuns şi nodul j din stratul de ieşire
$\mathbf{w}$	uneori identifică setul total de Π parametri / ponderi (atunci când nu este necesară utilizarea notaţiei separate $\mathbf{w}$, $\mathbf{v}$); $\mathbf{w} = [w_1, w_2,..., w_\Pi]^T$
Π	numărul total de parametri ai structurii
i	indice pentru elementele de intrare (perceptroni simpli din stratul de intrare, componentele vectorului de intrare; $1 \le i \le M$
h	indice pentru elementele stratului ascuns (perceptroni simpli din stratul ascuns, componentele vectorului $\mathbf{z}(\xi)$; $1 \le h \le H$
j	indice pentru elementele stratului de ieşire (perceptroni simpli din stratul de ieşire, componentele vectorului de ieşire obţinut, componentele vectorului de ieşire dorit; $1 \le j \le N$

Scanned with OKEN Scanner

3.2.2 Invăţarea în structurile neurale "feedforward"

3.2.2.1 Invăţarea în structurile "feedforward" ca proces de aproximare

Invăţarea reţelelor neurale "feedforward" (exemplu: perceptronul multistrat, harta auto – organizabilă, etc.), poate fi studiată ca proces de aproximare a unei funcţii $F : A \subset \mathcal{R}^M \rightarrow \mathcal{R}^N$, de la o mulţime mărginită a spaţiului euclidian M–dimensional la o submulţime mărginită $F(A)$ a spaţiului euclidian N – dimensional. Este rezolvată deseori folosind algoritmi supervizaţi, caz în care este disponibil setul de exemple de învăţare (x_1, d_1), (x_2, d_2), ... , (x_P, d_P), unde $d_p = F(x_p)$, $1 \le p \le P$. Modelul neural se mai numeşte "reţea de aproximare" ("mapping network"[8]). In continuare ne vom referi la învăţarea supervizată.

Din punctul de vedere menţionat, învăţarea este strâns legată de tehnicile clasice de aproximare, după cum au arătat Poggio şi Girosi [P3]. Teoria aproximării are ca obiect de studiu aproximarea sau interpolarea unei funcţii continue de mai multe variabile, $F(\mathbf{x})$, prin funcţia de aproximare $F(\mathbf{w}; \mathbf{x})$, cu un număr fixat Π de parametri $\mathbf{w}$. Pentru o anumită funcţie F, trebuie determinat setul de parametri $\mathbf{w}$ care realizează cea mai bună aproximare posibilă a funcţiei F pe setul de exemple. Există trei probleme şi anume:

❶ Ce aproximare se poate utiliza adică, ce clase de funcţii $F(\mathbf{x})$ pot fi aproximate efectiv de $F(\mathbf{w}; \mathbf{x})$. Aceasta este o problemă de reprezentare.

❷ Ce algoritm se poate utiliza pentru determinarea valorilor optime ale parametrilor $\mathbf{w}$, pentru un tip de funcţie F.

❸ Care este implementarea eficientă a algoritmului.

Prima dintre acestea este abordată în par. 3.2.2.1, iar cea de-a doua în par. 3.2.2.3. Cea de-a treia problemă nu reprezintă obiectul prezentei lucrări.

❒ ***Problema de aproximare***

Calitatea aproximării realizate de reţea în domeniul restrâns al setului datelor de învăţare, depinde de arhitectura reţelei utilizate şi de complexitatea problemei. Pentru a fi evaluată, se introduce o distanţă *dist* dintre funcţia $F(\mathbf{w}; \mathbf{x})$ (notată şi $F_{\mathbf{w}}$) şi funcţia $F(\mathbf{x})$, distanţă care este de obicei indusă de o normă (de exemplu, norma L_2). Dacă $F(\mathbf{x})$ este o funcţie continuă definită pe o mulţime $\{\mathbf{x}\}$, şi $F(\mathbf{w}; \mathbf{x})$ este o funcţie de aproximare ce depinde continuu de $\mathbf{w}$ şi $\mathbf{x}$, problema de aproximare constă în determinarea parametrilor $\mathbf{w}^*$ astfel încât, pentru toţi vectorii $\mathbf{w}$:

[8] Modul în care reţelele de tip "mapping networks" pot aproxima funcţii poate fi considerat ca o generalizare a analizei prin regresie statistică. In regresie, se alege mai întâi forma specifică a funcţiei ce urmează să aproximeze datele şi apoi se evaluează conform unui criteriu de cost ales (deseori eroare pătratică medie). Procedura a fost propusă de Gauss [H7].

Scanned with OKEN Scanner

$$\text{dist}[F(\mathbf{w}^*; \mathbf{x}), F(\mathbf{x})] \leq \text{dist}[F(\mathbf{w}; \mathbf{x}), F(\mathbf{x})] \quad (3.18)$$

O soluţie a acestei probleme se numeşte "cea mai bună aproximare", iar existenţa sa depinde de clasa de funcţii căreia îi aparţine $F(\mathbf{w}; \boldsymbol{x})$ [P3].

❒ *Problema de reprezentare. Aproximare universală*

Problema de reprezentare necesită precizarea aproximării care se poate utiliza (ce clase de funcţii $F(\boldsymbol{x})$ pot fi aproximate efectiv de $F(\mathbf{w}; \boldsymbol{x})$). In acest sens, reţelele neurale multistrat sunt modele neliniare ce realizează "aproximare universală": pot aproxima orice relaţie intrare – ieşire continuă, pe baza teoremelor datorate lui Kolmogorov, Cybenko, ş.a. Aceasta înseamnă că, familia de funcţii implementată de reţeaua neurală multistrat este suficient de mare pentru a conţine funcţia F sau o aproximare destul de bună a acesteia.

In anul 1957, A. Kolmogorov a publicat teorema sa referitoare la reprezentarea funcţiilor continue arbitrare definite pe cubul M – dimensional $I^M = [0, 1]^M$ cu valori în mulţimea numerelor reale $\mathcal{R}$, pentru funcţii de o singură variabilă. Ulterior au fost realizate o serie de îmbunătăţiri, printre care se numără cele propuse de G.G. Lorentz [H7].

Teorema 3.2 (Teorema lui Kolmogorov) Fiind dată orice funcţie continuă $F : [0, 1]^M \rightarrow \mathcal{R}^N$, $F(\boldsymbol{x}) = \boldsymbol{y}$, F poate fi implementată exact de o reţea neurală "feedforward" cu trei straturi având M elemente de prelucrare în primul strat, $(2M + 1)$ în cel de-al doilea şi respectiv N elemente de prelucrare în cele de-al treilea strat [H41].

Primul strat transmite semnalul de intrare mai departe, fără a realiza nici o prelucrare. Stratul ascuns implementează funcţia:

$$z_h = \sum_{i=1}^{M} \lambda^h \, \psi(x_i + h\,\varepsilon) + h \quad (3.19)$$

unde λ este o constantă reală, ψ este o funcţie continuă reală monoton crescătoare, independentă de F (deşi ambele depind de M), constanta ε este un număr raţional $0 < \varepsilon \leq \delta$, unde δ este o constantă pozitivă arbitrară.

Cele N elemente ale stratului de ieşire au valorile funcţiilor de activare date de relaţia (3.20), unde funcţiile g_j, $1 \leq j \leq N$ sunt reale şi continue (şi depind de F şi ε).

$$y_j = \sum_{h=1}^{2M+1} g_j(z_h) \quad (3.20)$$

Scanned with OKEN Scanner

Teorema lui Kolmogorov este o teoremă de existenţă, care afirmă că există o reţea neurală care să aproximeze funcţia F, dar nu precizează cum se construieşte această reţea (nu oferă exemple de funcţie ψ, nici valori ale constantei ε sau exemple de funcţii g). Cu toate acestea, implicaţiile pentru modelele neurale artificiale sunt deosebite, întrucât teorema oferă fundamentul teoretic pentru susţinerea ideii aproximării funcţiilor cu ajutorul RNA.

In anul 1965, Sprecher a completat teorema Kolmogorov:

Teorema 3.3 (Teorema Kolmogorov – Sprecher) Pentru fiecare număr întreg $M \geq 2$, există o funcţie reală monoton crescătoare $\psi(x)$, $\psi([0, 1]) = [0, 1]$, care depinde de M şi are proprietatea:

Pentru fiecare număr $\delta > 0$, există un număr raţional ε, $0 < \varepsilon < \delta$, astfel încât fiecare funcţie reală continuă $\varphi(x)$ de M variabile, definită pe cubul M–dimensional I^M poate fi reprezentată exact ca [K9]:

$$\varphi(x) = \sum_{h=1}^{2M+1} \chi\left[\sum_{i=1}^{M} \lambda^i \psi\left(x_i + \varepsilon(h-1)\right) + h - 1\right] \qquad (3.21)$$

unde χ este o funcţie reală şi continuă, care depinde de φ, iar λ este o constantă independentă de φ [K9].

După cum arăta Hecht – Nielsen, şi aceasta este o teoremă de existenţă, care indică faptul că orice funcţie reală continuă poate fi realizată exact de o reţea neurală compusă din patru straturi: stratul de intrare, două straturi ascunse şi un strat de ieşire, având toate număr finit de neuroni. Intrucât nu se precizează metode de determinare a valorii χ (care depinde de φ), aplicarea directă a teoremei este dificilă.

Cybenko [C502] a arătat în anul 1988 că, orice funcţie definită pe $\mathcal{R}^M$ cu valori în $\mathcal{R}^N$ poate fi implementată de o reţea neurală cu două straturi ascunse. Irie şi Miyake [K9] au demonstrat în acelaşi an că, o reţea cu trei straturi (un strat ascuns) şi număr infinit de noduri ascunse poate reprezenta o funcţie arbitrară, dacă funcţiile de activare ale nodurilor ascunse, precum şi funcţia intrare – ieşire sunt mărginite şi absolut integrabile. Hornik şi White [K7] au arătat în anul 1989 că, arhitecturile "feedforward" cu funcţii de activare sigmoidale pot reprezenta în principiu, cu orice acurateţe dorită, orice funcţie măsurabilă – Borel, dacă reţeaua conţine un număr suficient de neuroni în stratul / straturile ascunse. In acelaşi an, Hornik şi alţii [H503] au demonstrat că, un singur strat ascuns, compus dintr-un număr suficient de neuroni, garantează învăţarea oricărei funcţii, cu o acurateţe dată. Funahashi a extins în 1989 teorema Irie – Miyake, pentru a include funcţii de activare sigmoidale ale

Scanned with OKEN Scanner

nodurilor, astfel încât orice funcţie continuă este aproximativ realizată de o reţea cu trei straturi (un strat ascuns) şi funcţii de activare mărginite şi monoton crescătoare ale nodurilor ascunse. O abordare similară este realizată de Jones [J19], care porneşte de la observaţia că, orice funcţie continuă poate fi aproximată arbitrar de bine prin diferenţa dintre două funcţii continue strict crescătoare. Un rezultat similar celui arătat de Funahashi a obţinut Hecht–Nielsen, arătând că o subreţea a reţelei "feedforward" poate implementa o funcţie sinusoidală, permiţând reţelei să realizeze aproximarea prin serie Fourier a unei funcţii arbitrare [K9].

DasGupta şi Schnitger s-au referit la alte funcţii în afara celei sigmoidale, care permit menţinerea teoremelor de reprezentare existente. Ei au demonstrat că, funcţia de activare sigmoidală standard oferă reţelei o putere de aproximare comparabilă sau mai bună decât a altor clase de funcţii utilizate în teoria aproximării funcţiilor (ex.: spline, polinomiale) [D15, D19].

Este important de remarcat faptul că, teoremele de reprezentare neurală amintite sunt teoreme de existenţă. Acestea indică faptul că există o structură neurală "feedforward" care să implementeze funcţia F, dar nu precizează toate elementele necesare construirii modelului neural. De asemenea, deşi conform teoremelor amintite sunt suficiente trei straturi (un strat ascuns), în aplicaţii date utilizarea acestora poate conduce la un număr foarte mare de perceptroni simpli ascunşi, în timp ce două straturi ascunse pot reduce acest număr.

❒ *Eroare de învăţare şi eroare de test. Fenomenul de supraînvăţare*

Determinarea setului de parametri **w** ai reţelei neurale multistrat prin învăţare supervizată şi evaluarea aproximării realizate de aceasta se fac în două etape, şi anume învăţare propriu – zisă şi test. Prima se realizează pe setul de date de învăţare, conform unui algoritm dat, prin compararea valorii ieşirii obţinute cu cea dorită. Rezultatul său este setul de parametri (ponderi) ai modelului. Odată încheiată această fază, reţeaua este testată pe un set de date de test, diferit de cel de învăţare (de exemplu: date din setul de învăţare, cu zgomot), dar cu aceeaşi distribuţie a datelor. Dacă rezultatul testării este considerat corespunzător, atunci setul de parametri este rezultatul final al învăţării, altfel se reia învăţarea reţelei.

Eroarea obţinută în timpul învăţării este media pentru toate patternurile de intrare, a valorilor funcţiei cost, cu argument diferenţa dintre valoarea dorită şi cea obţinută la ieşire. Reprezentarea sa grafică se numeşte curbă de eroare. Aceasta se poate face în funcţie de numărul iteraţiilor sau de cel al epocilor. In cadrul unei iteraţii se prezintă un singur pattern de intrare reţelei, iar epoca reprezintă o trecere prin tot setul de patternuri. Dacă la fiecare iteraţie / epocă, caracterizată de o anumită valoare a erorii de învăţare, se testează reţeaua pe setul de test şi se reprezintă grafic, se obţine curba de test.

Scanned with OKEN Scanner

De obicei, pe parcursul primelor epoci, atât eroarea de învăţare cât şi cea de test au tendinţă descrescătoare, după care, deşi eroarea de învăţare continuă să scadă, cea de test începe să crească. Aceasta înseamnă că, deşi setul datelor de învăţare este aproximat din ce în ce mai bine de modelul neural, pentru date de test din afara acestuia, dar cu aceeaşi distribuţie, va fi incapabil să realizeze o aproximare corectă. Altfel spus, reţeaua neurală nu va putea generaliza (par. 3.2.3). Fenomenul de numeşte "supraînvăţare" ("overtraining", "overfitting") şi se poate datora următoarelor aspecte [R14]: ① datele de învăţare sunt incomplete, sau conţin neregularităţi, pe care reţeaua tinde să le înveţe; ② reţeaua are mai multe grade de libertate (parametri, în principal ponderi), decât numărul de eşantioanelor datelor de învăţare, deci nu există suficient de multe exemple pentru a impune constrângeri reţelei sau, ③ funcţia de aproximare $F(\mathbf{w}; x)$ realizată de reţea este foarte plată.

Ceea ce se urmăreşte este obţinerea unui set de parametri care să facă posibilă funcţionarea corespunzătoare a modelului neural într-un mediu în care există variaţii faţă de condiţiile de învăţare. Deci, i se cere structurii să generalizeze[9] pe baza setului de învăţare, pentru întregul set de test. De aceea, este importantă evitarea apariţiei fenomenului de supraînvăţare, prin oprirea învăţării înainte de apariţia acestuia. Printre metodele posibile, se pot menţiona: ❶ estimarea capacităţii de generalizare a reţelei în timpul învăţării şi oprirea învăţării atunci când aceasta începe să descrească; ❷ limitarea numărului de parametri ai reţelei (exemplu: prin reducerea numărului de noduri ascunse, deci şi a ponderilor), pentru a nu putea utiliza corelaţiile întâmplătoare dintre date [D205]. In plus faţă de generalizare mai bună, reţelele de dimensiuni mici au şi avantajul vitezei şi costului mai scăzut al implementării.

❒ *Compromisul deplasament – dispersie*

Presupunem că reţeaua "feedforward" realizează funcţia $F(\mathbf{w}; x)$, notată $F_{\mathbf{w}}(x)$, care aproximează funcţia $F(x)$. Scopul învăţării este minimizarea erorii de aproximare pentru toate exemplele posibile, deci nu numai pentru cele din setul de învăţare (altfel spus, reţeaua trebuie să generalizeze). Dacă eroarea este evaluată pe baza funcţiei cost eroare pătratică medie (EPM), atunci în orice punct x [K33]:

$$\mathrm{EPM}\{F_{\mathbf{w}}(\mathbf{x})\} = E\{(F(\mathbf{x}) - F_{\mathbf{w}}(\mathbf{x}))^2\} =$$

[9] In contextul teoriei aproximării, dacă datele de test fac parte din acelaşi interval cu cele de învăţare, deşi nu coincid cu acestea, atunci termenul "generalizare" poate fi înlocuit cu cel de "interpolare". Cu alte cuvinte, dacă un vector de date real (de test) se află aproape sau între datele de învăţare, atunci se doreşte obţinerea unei valori de ieşire a reţelei apropiată de cele obţinute pentru exemple din setul de învăţare. Dacă intrarea este departe de exemplele de învăţare, atunci ieşirea reţelei nu va avea nici o semnificaţie [H7]. Dacă datele de test nu fac parte din acelaşi domeniu cu cele de învăţare, dar au în continuare aceeaşi distribuţie, termenul de "generalizare se poate înlocui cu cel de "extrapolare".

Scanned with OKEN Scanner

$$= (F(x) - E\{F_w(x)\})^2 + E\{(F_w(x) - E[F_w(x)\})^2] =$$

$$= [\text{deplasament}(x)]^2 + \text{dispersie}(x) \qquad (3.22)$$

unde $E\{\bullet\}$ este operaţia matematică de mediere pe tot setul de date de învăţare de dimensiune P. Eroarea de test se obţine prin medierea tuturor valorilor EPM(x), pentru valorile x din spaţiul de intrare.

Deplasamentul şi dispersia se referă la funcţia $F_w(x)$, care aproximează $F(x)$. Obţinerea unui deplasament nul (estimat nedeplasat) şi unei dispersii mici caracterizează un estimat bun. Se observă că, în timpul învăţării reţelei, deplasamentul scade şi dispersia creşte (fig. 3.14). Este importantă realizarea compromisului dintre deplasament şi dispersie (cunoscut şi sub denumirea de compromis "bias – dispersie"), prin oprirea învăţării înainte ca eroarea de test să înceapă să crească [K33, M80].

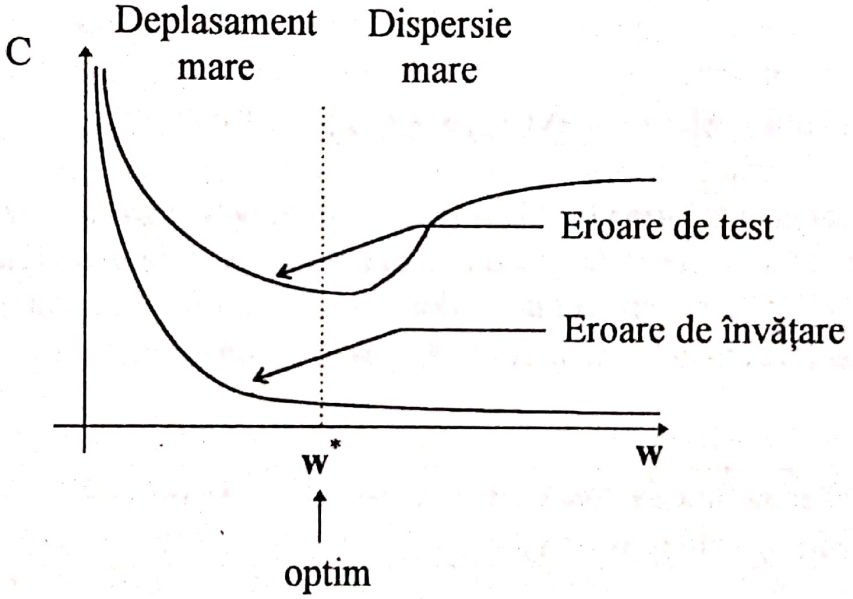


Fig. 3.14 Curbe tipice de eroare

3.2.2.2 Invăţarea în structurile "feedforward" ca proces de optimizare

Am precizat că, învăţarea constă în modificarea parametrilor **w** ai reţelei neurale printr-o strategie de optimizare corespunzătoare, în scopul determinării setului $\mathbf{w}^*$ care minimizează funcţia cost aleasă, pentru toate datele de învăţare. Această funcţie depinde de parametrii reţelei şi de datele amintite.

Determinarea ponderilor $\mathbf{w}^*$ se bazează pe căutarea în spaţiul tuturor configuraţiilor posibile de reţele. Pentru o arhitectură / problemă specifică, Judd [N6, A8] arată că, timpul necesar căutării depinde exponenţial de suma

dimensiunilor reţelei şi problemei studiate. In formularea adoptată de teoria complexităţii, se spune că, învăţarea în RNA este NP – completă.

Dacă funcţia cost $C(\mathbf{w})$ este aditivă şi pentru modelul neural se utilizează notaţiile din par. 3.2.1, atunci relaţia (3.17) devine:

$$\text{Minimizează funcţia cost } C(\mathbf{w}) = \sum_{\xi=1}^{P} \sum_{j=1}^{N} C_j\left(\mathbf{w}; \mathbf{x}(\xi)\right) = \sum_{\xi=1}^{P} C\left(\mathbf{w}; \mathbf{x}(\xi)\right)$$

$$\text{cu constrângerile } \mathbf{w} \in \Re^{\Pi} \qquad (3.23)$$

şi reprezintă formularea problemei estimării parametrilor $\mathbf{w}$, pentru cazul învăţării supervizate, cu $1 \le \xi \le P$ patternuri într-o structură cu $N > 1$ perceptroni simpli de ieşire. Condiţiile pe care trebuie să le satisfacă funcţia $C(\mathbf{w})$ sunt precizate de teorema 3.1. Pe de altă parte, funcţia $C(\mathbf{w}; \mathbf{x})$ trebuie să fie convexă:

$$C[(1-\xi)\,a + \xi\,b] \ge (1-\xi)\,C(a) + \xi\,C(b), \text{ cu } \xi \in [0, 1] \qquad (3.24)$$

Optimizarea sub forma (3.23) realizată cu modele neurale este prezentată în cazul general, atât cu constrângeri de tip egalitate cât şi de inegalitate de Cichoki şi Unbehauen [C16]. Pentru cel de-al doilea caz, aceasta se formulează ca (3.25), unde s-a presupus existenţa a q constrângeri şi Π parametri.

$$\text{Minimizează funcţia cost } C(\mathbf{w}) = C(w_1, w_2, \ldots w_{\Pi})$$
$$\text{cu constrângerile } g_i(\mathbf{w}) = g_i(w_1, w_2, \ldots w_{\Pi}) \ge 0, \quad \text{cu } 1 \le i \le q \qquad (3.25)$$

Pentru rezolvarea acesteia se scrie funcţia Lagrange (3.26), unde s-au notat cu λ_i multiplicatorii Lagrange.

$$L(\mathbf{w}, \lambda) = C(\mathbf{w}) + \sum_{i=1}^{q} \lambda_i\, g_i(\mathbf{w}) \qquad (3.26)$$

Presupunând că există o soluţie a problemei, notată cu $\mathbf{w}^* = \left[w_1^*, w_2^*, \ldots w_{\Pi}^*\right]$, pentru care $\min_{\mathbf{w}} C(\mathbf{w}) = C(\mathbf{w}^*)$, astfel încât toate constrângerile $g_i(\mathbf{w}^*) \ge 0$ sunt satisfăcute, se obţin condiţiile Kuhn – Tucker:

Scanned with OKEN Scanner

$$\frac{\partial C(\mathbf{w}^*)}{\partial w_j} + \sum_{i=1}^{q} \lambda_i^* \frac{\partial g_i(\mathbf{w}^*)}{\partial w_i} = 0 \qquad 1 \le j \le \Pi \tag{3.27}$$

$$g_i(\mathbf{w}^*) \ge 0 \qquad 1 \le i \le q \tag{3.28}$$

$$\lambda_i^* \le 0 \qquad 1 \le i \le q \tag{3.29}$$

$$\lambda_i^* g_i(\mathbf{w}^*) = 0 \qquad 1 \le i \le q \tag{3.30}$$

3.2.2.3 Algoritmi de învăţare în structurile "feedforward"

Există o serie de algoritmi care pot fi utilizaţi pentru rezolvarea problemelor de optimizare. Soluţia (vectorul optimal $\mathbf{w}^*$ al parametrilor reţelei, care minimizează funcţia cost) nu este unică, datorită permutărilor posibile ale ponderilor şi nodurilor ascunse într-o reţea "feedforward". Astfel, spaţiul Ω al ponderilor ($\mathbf{w} \in \Omega$), constă din mai multe conuri identice, fiecare conţinând un minim global. In afara punctelor de minim global pot exista puncte de minim local şi este importantă evitarea blocării algoritmului datorită acestora. De obicei, se utilizează evaluări ale funcţiei cost şi derivatelor sale, pentru a localiza minimul (local sau global) şi de a opri căutarea în spaţiul ponderilor în punctul respectiv.

Pentru determinarea soluţiei problemei de optimizare, este necesară precizarea tipului algoritmului (determinist sau stohastic), direcţiei de căutare a minimului în spaţiul ponderilor, strategiei de alegere a parametrilor (rată de învăţare, moment, etc.) şi tipului metodei (modificare pas cu pas vs. modificare de tip bloc) [K36, S50].

- ***Determinarea direcţiei de căutare a minimului în spaţiul parametrilor***

Fie $\mathbf{w} = [w_1, ..., w_\Pi]^T$ vectorul tuturor celor Π parametri ai reţelei (ponderi, praguri). Presupunem că, determinarea minimului funcţiei cost $C(\mathbf{w})$ în spaţiul parametrilor se realizează în timp discret, prin căutare liniară. Aceasta porneşte din punctul iniţial $\mathbf{w}_0$ în direcţia $\boldsymbol{d}$ (notaţie care nu trebuie confundată cu cea utilizată pentru vectorul valorilor de ieşire dorite $\boldsymbol{d}$ din reţeaua neurală "feedforward"), şi urmăreşte segmentele de dreaptă definite la fiecarea pas k de relaţia (3.31). S-a notat cu η_k pasul (rata de învăţare) la momentul k.

Scanned with OKEN Scanner

Vectorii de ponderi $\mathbf{w}(k)$ şi respectiv $\mathbf{w}(k+1)$ la momentele de timp discret k şi respectiv $k+1$, au fost notaţi cu $\mathbf{w}_k$ şi $\mathbf{w}_{k+1}$ [K19, H32]:

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \eta_k \mathbf{d}_k \tag{3.31}$$

Determinarea direcţiei de căutare $\mathbf{d}_k$ se poate face prin metode deterministe sau stohastice. Primele se numesc metode de ordinul întâi dacă utilizează derivatele de ordinul întâi ale funcţiei cost, respectiv de ordinul doi dacă utilizează derivatele de ordinul doi. Metodele stohastice pot fi de tip "simulated annealing" [L41, H3] sau metode bazate pe algoritmi genetici [S11, B2].

❒ *Metode deterministe de ordinul întâi*

Metodele deterministe de ordinul întâi folosesc numai valoarea gradientului local pentru aflarea direcţiei de minimizare. Se mai numesc metode de gradient negativ ("steepest descent", "gradient descent"). In reţelele "feedforward", acestea stau la baza algoritmului BKP de propagare inversă a erorii [R3, ş.a.]. Metoda gradientului negativ constă în minimizarea funcţiei cost în lungul dreptei $\mathbf{d}_k$, în direcţia gradientului. Gradientul este re–evaluat şi procesul se repetă.

$$\mathbf{d}_k = -\nabla C(\mathbf{w}_k) \tag{3.32}$$

Paşii succesivi realizaţi în direcţia gradientului negativ sunt perpendiculari, deoarece relaţia (3.33) implică perpendicularitatea noii direcţii a gradientului $\nabla C(\mathbf{w}_{k+1})$ pe vechea direcţie $\mathbf{d}_k$, după minimizarea liniară. Astfel, rezultă o traiectorie în zigzag.

$$\mathbf{0} = \frac{\partial}{\partial \eta_k} C(\mathbf{w}_k + \eta_k \mathbf{d}_k) = \mathbf{d}_k \nabla C(\mathbf{w}_{k+1}) \tag{3.33}$$

Metoda gradientului negativ este una dintre cele mai simple metode de optimizare [K19, N20]. Este rapidă, dar fiecare pas presupune o minimizare liniară, care poate consta din evaluarea mai multor funcţii.

Pe de altă parte, metoda nu este performantă, în sensul că algoritmul se poate bloca în puncte de minim local. Viteza de convergenţă este scăzută.

❒ *Metode deterministe de ordinul doi*

Metodele deterministe de ordinul doi utilizează şi derivatele de ordinul doi ale funcţiei cost $C(\mathbf{w})$. Printre acestea se numără metoda gradientului conjugat, metoda Newton, metodele cvasi– şi pseudo – Newton.

Scanned with OKEN Scanner

■ Metoda gradientului conjugat (I)

In metoda gradientului conjugat, propusă de Fletcher şi Reeves, ideea de bază este realizarea unui compromis între direcţia gradientului şi direcţia de căutare anterioară, pentru determinarea noii direcţii de căutare (3.34) în spaţiul parametrilor. Direcţia iniţială este (3.35).

$$\mathbf{d}_k = -\nabla C(\mathbf{w}_k) + \beta_k \mathbf{d}_{k-1} \tag{3.34}$$

$$\mathbf{d}_0 = -\nabla C(\mathbf{w}_0) \tag{3.35}$$

Din relaţia (3.34) se observă că, metoda gradientului conjugat este un caz particular al metodei gradientului negativ cu moment (3.36), în care momentul $\alpha = \beta_k$.

$$\mathbf{d}_k = -\nabla C(\mathbf{w}_k) + \alpha \mathbf{d}_{k-1} \tag{3.36}$$

Valoarea β_k se alege astfel încât fiecare nouă direcţie de căutare să afecteze cât mai puţin minimizarea realizată în direcţia anterioară. Astfel, direcţia de căutare $\mathbf{d}_k$ nu trebuie să modifice componentele gradientului în lungul direcţiei anterioare $\mathbf{d}_{k-1}$, care a fost anulată conform relaţiei (3.33). De aceea, până la prima derivată în raport cu η_k, trebuie să fie valabilă relaţia:

$$\mathbf{d}_{k-1} \nabla C(\mathbf{w}_k + \eta \mathbf{d}_k) = 0 \tag{3.37}$$

Funcţia cost $C(\mathbf{w})$, în jurul punctului curent $\mathbf{w}_k$, poate fi aproximată prin seria Taylor trunchiată:

$$C(\mathbf{w}) = C(\mathbf{w}_k) + (\mathbf{w} - \mathbf{w}_k)^T \nabla C(\mathbf{w}_k) + \frac{1}{2}(\mathbf{w} - \mathbf{w}_k)^T \nabla^2 C(\mathbf{w}_k) (\mathbf{w} - \mathbf{w}_k) \tag{3.38}$$

Notăm cu $\boldsymbol{J}(\mathbf{w}_k)$ matricea iacobiană (matricea derivatelor de ordinul întâi ale funcţiei cost) şi cu $\boldsymbol{H}(\mathbf{w}_k)$ matricea hessiană (matricea derivatelor de ordinul doi ale funcţiei cost), ambele evaluate în punctul $\mathbf{w}_k$:

$$\mathbf{J}(\mathbf{w}_k) = \nabla C(\mathbf{w}_k) = \left.\frac{\partial C}{\partial \mathbf{w}_r}\right|_{\mathbf{w}_k} \tag{3.39}$$

Scanned with OKEN Scanner

$$\mathbf{H}(\mathbf{w}_k) = \nabla^2 C(\mathbf{w}_k) = \left. \frac{\partial^2 C}{\partial w_r \, \partial w_s} \right|_{\mathbf{w}_k} \tag{3.40}$$

Prin diferenţierea expresiei (3.38) şi utilizarea notaţiilor (3.39) – (3.40), se obţine relaţia (3.41). Folosind relaţiile (3.33) şi (3.41), expresia (3.37) se poate rescrie ca (3.42). Vectorii $\mathbf{d}_{k-1}$ şi $\mathbf{d}_k$ se numesc vectori conjugaţi [K19].

$$\nabla C(\mathbf{w}) = \mathbf{J}(\mathbf{w}_k) + \mathbf{H}(\mathbf{w}_k)\,(\mathbf{w} - \mathbf{w}_k) + \cdots \tag{3.41}$$

$$\mathbf{d}_{k-1}\,\mathbf{H}\,\mathbf{d}_k = 0 \tag{3.42}$$

Alegerea valorii β_k din expresia (3.34), pentru satisfacerea (3.42) se poate face în funcţie de problema dată, conform uneia dintre relaţiile Fletcher – Reeves, Polack – Ribiere, Hestenes – Stiefel [C16].

Metodele de gradient conjugat au fost utilizate în rezolvarea multor probleme cu reţele neurale, cu rezultate bune. Avantajul faţă de metoda gradientului negativ este viteza de convergenţă ridicată (convergenţa teoretică se realizează în Π paşi, cu Π numărul parametrilor structurii neurale), precum şi simplitatea. Nu este necesară inversarea unor matrici hessiene de mari dimensiuni [H32].

In practică, este deseori necesară pornirea periodică a procesului de optimizare, datorită rezultatelor lipsite de acurateţe obţinute. Acestea se pot datora problemei care poate să nu fie pătratică (caz în care aproximaţia realizată prin dezvoltarea în serie până la termeni pătratici nu este utilă). Imbunătăţirea metodelor de gradient conjugat se face prin alegerea parametrului β şi prin diferite proceduri de restart (exemplu: metoda de restart Powell, prezentată de Van der Smagt [S50]).

■ Metoda Newton (II)

Metoda Newton se bazează pe ideea că, funcţia cost ce trebuie minimizată poate fi aproximată local printr-o funcţie pătratică, iar aceasta poate fi minimizată exact. Funcţia cost $C(\mathbf{w})$ în jurul punctului curent $\mathbf{w}_k$ poate fi aproximată prin seria Taylor trunchiată (3.38). Cu notaţiile (3.39) – (3.40), aceasta devine prin diferenţiere (3.41). Minimul funcţiei cost $C(\mathbf{w})$, ce reprezintă soluţia problemei de optimizare, este un punct în care, evident, $\nabla C(\mathbf{w}) = 0$ şi se poate determina prin egalarea cu zero a relaţiei (3.41) şi ignorarea termenilor de ordin superior ai dezvoltării [C16, K19, H20]:

Scanned with OKEN Scanner

$$\mathbf{J}(\mathbf{w}_k) + \mathbf{H}(\mathbf{w}_k)\,(\mathbf{w} - \mathbf{w}_k) = \mathbf{0} \tag{3.43}$$

Dacă inversa matricii hessiene există, atunci se obţine relaţia lui Newton (3.44) în timp discret. Aceasta poate fi utilizată iterativ, considerând estimatul $\mathbf{w}$ obţinut, drept noua valoare $\mathbf{w}_k$ pentru pasul următor şi se obţine expresia (3.45).

$$\mathbf{w} = \mathbf{w}_k - \mathbf{H}^{-1}(\mathbf{w}_k)\,\mathbf{J}(\mathbf{w}_k) \tag{3.44}$$

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \mathbf{H}^{-1}(\mathbf{w}_k)\,\mathbf{J}(\mathbf{w}_k) \tag{3.45}$$

Metoda Newton are avantajul convergenţei rapide. De asemenea, nu se blochează în puncte de minim local.

Printre dezavantajele metodei, se numără:

❶ Metoda Newton este foarte costisitoare, deoarece pentru Π ponderi trebuie calculate inversele matricii $\boldsymbol{H}$ de dimensiuni Π x Π la fiecare pas k, durata fiind de fiecare dată de ordinul Π^3 paşi. O soluţie pentru rezolvarea acestei probleme este utilizarea metodelor de aproximare a inversei matricii hessiene.

❷ Matricea hessiană poate fi singulară

❸ Metoda necesită calculul derivatelor de ordinul doi

❹ Dacă optimizarea porneşte dintr-un punct iniţial apropiat de minim (situaţie puţin probabilă în cazurile reale), atunci se obţin rezultate bune. Dacă punctul iniţial este departe de minimul local, atunci este posibil ca matricea hessiană să nu fie pozitiv definită şi algoritmul Newton este divergent.

Algoritmul Newton standard pentru structuri "feedforward" a fost studiat de Diniz, Campos şi Antoniou [D32], ş.a. Există numeroşi alţi algoritmi ce se bazează pe metoda Newton, printre care algoritmul Quickprop propus de Scott Fahlman în anul 1988 [F1, F18].

■ Metode cvasi – Newton (III)

Metodele cvasi – Newton, sau metode cu metrică variabilă, urmăresc determinarea punctului de minim pe o suprafaţă de cost pătratică în Π paşi. Ideea este utilizarea metodei Newton, dar cu aproximarea inversei matricii hessiene $\boldsymbol{H}$, pentru micşorarea volumului de calcul pe care îl implică calculul explicit al acesteia.

Se notează cu $\hat{\boldsymbol{H}}^{-1}$ matricea ce aproximează inversa matricii hessiene $\boldsymbol{H}^{-1}$. Indicele k marchează pasul de timp discret k. Relaţia Newton în care se utilizează $\hat{\boldsymbol{H}}^{-1}$ se scrie:

Scanned with OKEN Scanner

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \eta_k \, \mathbf{d}_k = \mathbf{w}_k - \eta_k \, \hat{\mathbf{H}}_k^{-1} \, \nabla C(\mathbf{w}_k) \tag{3.46}$$

Valoarea ratei de învăţare η_k în pasul k se alege prin minimizarea liniară (3.47), iar aproximarea inversei matricii hessiene este dată de relaţia (3.48).

$$\eta_k = \arg \min_{\eta \geq 0} C\left(\mathbf{w}_k - \eta \, \hat{\mathbf{H}}_k^{-1} \, \nabla C(\mathbf{w}_k)\right) \tag{3.47}$$

$$\hat{\mathbf{H}}_{k+1}^{-1} = \hat{\mathbf{H}}_k^{-1} + \Psi\left(\hat{\mathbf{H}}_k^{-1}, \mathbf{w}_{k+1} - \mathbf{w}_k, \nabla C(\mathbf{w}_{k+1}) - \nabla C(\mathbf{w}_k)\right) \tag{3.48}$$

Funcţia Ψ nu are o formă simplă. Relaţia (3.48) realizează aproximaţii succesive (pornind de la matricea unitate) ale matricii hessiene, fără a evalua însă vreo derivată de ordinul doi [K19].

Există mai multe metode din categoria cvasi – Newton, printre care metoda Broyden – Fletcher – Goldfarb – Shanno, Davidon – Fletcher – Powell, Barnes–Rosen, Levenberg – Marquardt.

In metodele cvasi – Newton se evită calculul derivatelor de ordinul doi. Sunt metode stabile, de vreme ce direcţia de căutare $\mathbf{d}$ este totdeauna cea a gradientului negativ. Rezultatele sunt comparabile cu cele ale metodei gradientului conjugat. Viteza metodelor cvasi – Newton este similară celei a gradientului conjugat, dar acestea necesită mai multă memorie, datorită matricii $\hat{\boldsymbol{H}}^{-1}$. Spaţiul necesar stocării matricii hessiene creşte pătratic cu numărul ponderilor structurii neurale.

■ Metoda pseudo – Newton (IV)

Metoda pseudo – Newton se bazează pe ignorarea elementelor din afara diagonalei principale a matricii hessiene $\boldsymbol{H}$. Legea de modificare a ponderilor este:

$$\Delta w_{ij} = -\frac{\partial C / \partial w_{ij}}{\partial^2 C / \partial w_{ij}^2} \tag{3.49}$$

Metoda este echivalentă aplicării regulii Newton separat, pentru fiecare pondere. Dacă numitorul este negativ sau tinde la zero, aplicarea directă a relaţiei (3.49) poate conduce la realizarea unor paşi mari în direcţie greşită. Evitarea acestei situaţii se poate face prin utilizarea relaţiei modificate (3.50), cu μ o constantă mică pozitivă.

Scanned with OKEN Scanner

$$\Delta w_{ij} = -\frac{\partial C / \partial w_{ij}}{\left|\frac{\partial^2 C}{\partial w_{ij}^2}\right| + \mu} \qquad (3.50)$$

❒ *Metode stohastice*

O abordare diferită de cea deterministă este cea bazată pe algoritmi genetici. Aceasta presupune căutarea în spaţiul ponderilor, fără utilizarea informaţiilor legate de gradient [B2, H3]. Algoritmii genetici realizează o căutare globală în spaţiul ponderilor, de aceea nu se blochează în puncte de minim local. Deseori, căutarea genetică iniţială este urmată de aplicarea metodei gradientului negativ. Deşi eficienţi, algoritmii genetici sunt consumatori de timp şi de resurse, datorită explorării întregului spaţiu al ponderilor.

Un alt tip de metodă stohastică se bazează pe determinarea punctelor de minim global ale funcţiei cost $C(\mathbf{w})$ prin metoda "simulated annealing" (SA) [L41, H19].

❒ *Concluzii privind metodele de căutare prezentate*

Am prezentat metode deterministe şi am amintit metode stohastice de căutare a minimului funcţiei cost în spaţiul parametrilor. Din prima categorie, metodele de ordinul doi au avantajul convergenţei sigure, precum şi al renunţării la "ghicirea" valorilor iniţiale ale parametrilor, în comparaţie cu metodele de ordinul întâi. Pe de altă parte, volumul de calcul creşte, deci şi cerinţele impuse hardware-ului / software-ului. Dintre acestea, metoda gradientului conjugat poate fi considerată din punct de vedere al performanţelor şi complexităţii, o variantă intermediară între metoda gradientului negativ (de ordinul întâi) şi metodele cvasi – Newton. O comparaţie a metodelor gradientului negativ şi gradientului conjugat a fost realizată de Kung [K36]. S-a sugerat, de asemenea, aplicarea unor algoritmi hibrizi, în care metoda de ordinul întâi cu rată de învăţare fixă să fie înlocuită în apropierea punctului de minim, metode de ordinul doi.

Metodele stohastice nu utilizează informaţii legate de gradient sau de derivatele de ordinul doi ale funcţiei cost. Au viteză scăzută şi o complexitate ridicată a calculelor, datorită căutării în întregul spaţiu al parametrilor, dar rezultatele obţinute sunt bune.

Scanned with OKEN Scanner

3.2.3 Generalizarea în structurile neurale "feedforward"

Sistemele inteligente generalizeaza, de asemenea. Diversitatea comportamentului întrece experienta acumulata. Filozoful secolului al 18lea, David Hume, a vazut de ce: "Sistemele inteligente asociaza raspunsuri similare unor stimuli similari" (Bart Kosko).

Traim într-o lume de o complexitate si varietate minunata, o lume în care evenimentele nu se repeta niciodata cu exactitate. Heraclit comenta, acum peste doua milenii si jumatate ca, "niciodata nu pasim de doua ori în acelasi râu". Dar, desi evenimentele nu sunt exact aceleasi, nici nu sunt complet diferite. Exista un fir de continuitate, de similaritate si predictibilitate care ne permite sa generalizam, deseori corect, din experienta anterioara, pentru evenimente viitoare (idem).

Odată încheiat procesul de învăţare, structura neurală este caracterizată de un set de parametri (ponderi, praguri) cu valori fixe. Reţeaua este testată pe un set de date diferit de cel de învăţare. Scopul final este minimizarea erorii pe setul de date de test, ceea ce implică generalizarea cunoştinţelor acumulate de modelul neural pe setul de învăţare.

Fără capacitatea de generalizare, RNA ar suscita un interes limitat. De aceea, se impune evaluarea modelului neural şi din acest punct de vedere, nu numai din cel al dimensiunilor structurii şi vitezei de convergenţă în procesul de învăţare.

Definiţia 3.22 Generalizarea este abilitatea de a estima cantitativ caracteristicile unui fenomen care nu a fost întâlnit anterior, pe baza similitudinilor sale cu alte fenomene deja cunoscute [B23, L34].

Această abilitate implică faptul că, modelul care generalizează poate separa caracteristici similare a două fenomene şi poate, de asemenea, să distingă trăsăturile specifice de cele generale [A502, K501]. Este important să se precizeze ce anume aşteaptă proiectantul de la reţeaua neurală, pentru a putea afirma că aceasta generalizează. Presupunem că RNA învaţă să aproximeze o funcţie F, fiind date seturile de puncte $(x, F(x))$. Pentru valori ale variabilei x care fac parte din intervalul pe care l-a învăţat, funcţia trebuie să realizeze interpolarea. Pentru puncte din afara acestuia, va realiza extrapolarea. Există un mare număr de generalizări posibile, de aceea pentru a stabili care este cea preferată de proiectant, trebuie stabilit un criteriu de apreciere a generalizării.

O a doua observaţie este aceea că, dacă a învăţat prea bine setul de date, are loc fenomenul de "supraînvăţare", care conduce la performanţe slabe ale modelului neural, ca interpolator sau extrapolator.

Scanned with OKEN Scanner

Pe baza observaţiilor anterioare, s-au formulat o serie de probleme în cadrul teoriei generalizării, printre care se numără:

- Ce proprietăţi ale reţelei neurale şi ale aplicaţiei pe care trebuie să o rezolve, determină generalizarea cu succes ?
- Cât de dificilă este învăţarea unei probleme date de către reţeaua neurală dată ?
- Ce criterii de evaluare a capacităţii de generalizare se pot alege ?
- Cum se poate îmbunătăţi generalizarea ?
- Care este numărul necesar de patternuri de învăţare pentru obţinerea unei generalizări bune ?

La prima dintre acestea au răspuns Denker ş.a. [D200], Levin, Tishby şi Sola [L34], care au studiat problema generalizării din punct de vedere statistic, în cadrul teoriei învăţării. Valiant a propus în anul 1984 un model de studiu al generalizării. Alte studii folosesc conceptul de dimensiune Vapnik–Chervonenkis, pentru a determina cazurile cele mai defavorabile pentru modelul neural, în care se obţin limite minime ale generalizării sau, se concentrează asupra valorilor la care tinde asimptotic eroarea medie de generalizare a sistemului neural.

Deşi au stabilit condiţiile generale necesare obţinerii generalizării bune, nici una dintre aceste abordări nu a răspuns întrebării legate de dificultatea învăţării unei probleme date de către RNA. S-a precizat însă că, este necesară selectarea corectă a setului de învăţare, a parametrilor de control al învăţării şi a dimensiunilor reţelei.

3.2.3.1 Evaluarea capacităţii de generalizare

Generalizarea poate fi apreciată conform următoarelor criterii [S24]: ❶ probabilitatea ca reţeaua care a învăţat să genereze, în medie, ieşirea corectă pentru o valoare de intrare aleasă aleator; ❷ probabilitatea ca reţeaua care a învăţat să genereze, în cazul cel mai defavorabil, ieşirea corectă pentru o valoare de intrare aleasă aleator; ❸ numărul mediu de generalizări alternative pentru setul de învăţare dat.

In cazul ❶ se realizează medierea peste toate reţelele posibile care sunt consistente cu (pot prelucra) setul de învăţare. In cazul ❷ se alege cea mai slabă reţea din punct de vedere al performanţei (dintre cele consistente cu setul de învăţare), astfel încât aceasta să furnizeze o limită a probabilităţii de eroare. In fiecare caz se poate utiliza măsura generalizării pentru estimarea dimensiunilor setului de date de învăţare, pentru o generalizare "bună", definind "bun" prin îndeplinirea condiţiei ca eroarea probabilă să fie mai mică decât un număr convenabil.

Varianta ❶ a fost studiată de Hertz, Krogh şi Palmer [H19], care au abordat teoretic generalizarea, evaluată ca probabilitate ca reţeaua care a învăţat, să

Scanned with OKEN Scanner

genereze în medie, ieşirea corectă pentru o intrare aleasă aleator. Ideea de bază este evaluarea reducerii volumului scanat din spaţiul ponderilor (în care un vector **w** reprezintă un punct), datorită învăţării celor P exemple date. Pe baza acestora se poate calcula capacitatea de generalizare, care depinde numai de arhitectură şi de constrângerile apriori asupra ponderilor. Evaluarea generalizării medii permite apoi, predicţia numărului de exemple necesare pentru învăţarea reţelei, în scopul obţinerii unei performanţe medii dorite.

Determinarea generalizării medii se poate face prin estimarea erorii medii obţinute pe setul de test (eroare de generalizare medie). Funcţia cost C este de cele mai multe ori, o sumă de doi termeni: eroarea de învăţare C_{inv} (numită şi termen standard) şi un termen de regularizare C_r (numit de penalizare datorată complexităţii structurii). Cel din urmă se mai numeşte eroare de generalizare (C_g) sau de test.

$$C = C_{inv} + C_g \tag{3.51}$$

Din relaţia (3.51) rezultă că, pentru minimizarea funcţiei cost C este necesară atât minimizarea erorii de învăţare C_{inv}, cât şi a erorii de generalizare. Dacă valoarea celei din urmă nu prezintă interes, atunci se pot aplica metode de interpolare care garantează faptul că eroarea de aproximare pe setul de învăţare C_{inv} poate fi anulată. O astfel de metodă este de exemplu, metoda "one sample one unit" [H505, H30].

Eroarea medie de generalizare Γ se constituie din trei componente [L33, L34]:

$$\Gamma = \sigma^2_{zgomot} + EPM + PFP \tag{3.52}$$

unde σ^2_{zgomot} este dispersia zgomotului existent în datele de test, EPM este eroarea pătratică medie de învăţare, iar PFP este termenul de penalizare datorat fluctuaţiei ponderilor. Obţinerea unei valori Γ minime se realizează prin compromisul dintre ultimii doi termeni: EPM poate fi redusă prin învăţare sau creşterea complexităţii modelului (numărului de parametri), iar PFP de obicei creşte în timpul învăţării.

Se consideră modelul neural:

$$d(\xi) = F(\mathbf{w}; \mathbf{x}(\xi)) + e(\mathbf{w}; \xi) \tag{3.53}$$

unde $F(\mathbf{w}; \bullet)$ este funcţia realizată de reţea (notată şi $F_w(x)$) având ca parametri setul Π – dimensional de ponderi **w**. Fără a restrânge generalitatea, s-a presupus existenţa unui singur nod de ieşire $N = 1$ şi s-a notat cu $e(\mathbf{w}; \xi)$ semnalul de

Scanned with OKEN Scanner

eroare obţinut în acest nod, pentru patternul de intrare ξ. Se consideră setul de date de învăţare cu P patternuri, $\Delta = \{x(\xi); d(\xi)\}$, $1 \le \xi \le P$. Parametrii modelului (ponderi, praguri) sunt estimaţi prin minimizarea funcţiei cost, ai cărei termeni se pot scrie conform relaţiei:

$$C_P(\mathbf{w}) = C_{inv}(\mathbf{w}) + \mathbf{w}^T \mathbf{R}\, \mathbf{w} \tag{3.54}$$

unde $\boldsymbol{R}$ este o matrice pozitiv semidefinită, cu dimensiunile Π x Π (numită matrice de regularizare, par. 3.2.4), iar $(\bullet)^T$ marchează operaţia de transpunere. S-a notat cu C_{inv} expresia:

$$C_{inv}(\mathbf{w}) = \frac{1}{P}\sum_{\xi=1}^{P} e^2(\mathbf{w};\xi) = \frac{1}{P}\sum_{\xi=1}^{P} \left[d(\xi) - F(\mathbf{w};\mathbf{x}(\xi))\right]^2 \tag{3.55}$$

Ponderile $\hat{\mathbf{w}}$ estimate prin învăţare sunt numite ponderi – estimat (relaţia (3.56)). Valoarea medie a funcţiei cost este (3.57), unde $E\{\bullet\}$ marchează operaţia de mediere în raport cu funcţia densitate de probabilitate intrare – ieşire.

$$\hat{\mathbf{w}} = \arg \min_{\mathbf{w}} C_P(\mathbf{w}) \tag{3.56}$$

$$C(\mathbf{w}) = E\{C_P(\mathbf{w})\} = E\{e^2(\mathbf{w})\} + \mathbf{w}^T \mathbf{R}\, \mathbf{w} \tag{3.57}$$

Dacă se respectă condiţiile impuse, atunci $lim_{P\to\infty} C_P(\mathbf{w}) = C(\mathbf{w})$ şi vectorul de ponderi estimat $\hat{\mathbf{w}}$ devine un estimat consistent ($P \to \infty$) al vectorului de ponderi optimal: $\mathbf{w}^* = arg\ min_{\mathbf{w}} C(\mathbf{w})$. Eroarea de generalizare a modelului estimat este definită ca eroarea pătratică medie pe setul de test, independent de patternurile de învăţare [L33, L34]:

$$G(\hat{\mathbf{w}}) = E\left\{e^2(\hat{\mathbf{w}})\right\} = E\left\{\left[d - F(\hat{\mathbf{w}};\mathbf{x})\right]^2\right\} \tag{3.58}$$

Conform relaţiei (3.58), $G(\hat{\mathbf{w}})$ depinde de setul de învăţare, prin intermediul valorilor ponderilor estimate $(\hat{\mathbf{w}})$, deci nu este o măsură foarte bună a estimării realizate a parametrilor. In plus, nu este posibilă obţinerea valorilor estimate $G(\hat{\mathbf{w}})$, fără a cunoaşte distribuţia intrare – ieşire. De aceea, este mai utilă evaluarea erorii medii de generalizare:

$$\Gamma = E_{seturi\ P}\left\{G(\hat{\mathbf{w}})\right\} \tag{3.59}$$

Scanned with OKEN Scanner

unde medierea se face pe toate seturile de învăţare, având fiecare P patternuri. Estimatorului erorii medii de generalizare Γ, notat cu $\hat{\Gamma}$ şi obţinut pe baza relaţiei (3.59), i se pot impune din punct de vedere statistic "cerinţe de calitate". De exemplu:

Definiţia 3.23 Estimatorul căutat $\hat{\Gamma}$ trebuie să fie consistent ($\hat{\Gamma} \to \Gamma$, când $P \to \infty$) şi nedeplasat până la ordinul $1/P$ (adică, $E_{\text{seturi P}}\left\{\hat{\Gamma}\right\} = \Gamma + O(1/P)$) [L33].

Evaluarea capacităţii medii de generalizare prin medierea peste o regiune din spaţiul ponderilor (varianta ❷) are dezavantajul că, o anumită regiune poate fi favorizată de legea de învăţare. Când se parcurge un set de învăţare dat, se urmăreşte de fapt, o anumită traiectorie în spaţiul ponderilor, pe măsură ce acestea sunt modificate. Altfel spus, nu se realizează selecţia aleatoare a unui set de ponderi cu constrângerea ca reţeaua să înveţe setul datelor de învăţare. De aceea, trebuie luată în considerare evaluarea capacităţii de generalizare în cazul cel mai defavorabil, şi nu în cazul mediu.

3.2.3.2 Imbunătăţirea capacităţii de generalizare

Relaţia dintre eroarea de învăţare şi cea de test, reprezentată grafic în fig. 3.14, pune în evidenţă apariţia fenomenului de supraînvăţare. Am menţionat anterior că, evitarea acestuia conduce la îmbunătăţirea generalizării. Se poate realiza prin una dintre metodele prezentate în continuare.

❒ ***Estimarea capacităţii de generalizare a reţelei în timpul învăţării şi oprirea învăţării când aceasta începe să descrească*** **(I)**

❶ Impărţirea setului de date într-un set de învăţare şi un set de test

In mod ideal, învăţarea şi testarea ar trebui realizate pe seturi de date de dimensiune infinită. Intrucât acest lucru nu este posibil, trebuie să se verifice că acestea sunt suficient de mari. In cazul în care creşterea setului nu conduce la modificarea performanţelor, condiţia amintită este îndeplinită. O altă soluţie de verificare este partajarea setului de date într-un subset de învăţare şi un subset de test (de validare) şi de a arăta că, performanţele pe fiecare dintre acestea sunt apropiate [H7]. Este esenţial ca subsetul de învăţare să fie utilizat numai în etapa de învăţare, iar cel de validare numai în etapa de test.

Deşi există structuri neurale care nu suferă de fenomenul de supraînvăţare (dintre cele "feedforward" menţionăm hărţile auto – organizabile), deci pentru

Scanned with OKEN Scanner

care nu este utilă existenţa unui set de învăţare diferit de cel de test (învăţarea este oprită când curba eroare de învăţare coboară sub un nivel acceptabil), în continuare ne vom referi la modelele "feedforward" în care apare acest fenomen. Printre metodele de minimizare a erorii de test se numără validarea încrucişată ("cross – validation"), metoda "leave one out", etc.

❷ Estimarea teoretică a generalizării, pe baza performanţei înainte şi după învăţare

Estimarea erorii de generalizare se poate face prin relaţii empirice sau pe baza studiilor teoretice în domeniul generalizării, care utilizează informaţii legate de performanţa înainte şi după învăţare [L34]. Lee şi Tenorio [H30] au propus minimizarea erorii empirice minime generalizate EMGM (criteriul "Generalized Minimum Empirical Error"). Akaike a propus criteriul numit "Final Prediction Error" pentru estimarea erorii de generalizare cunoscând eroarea de învăţare (par. 4.4.3). Moody a modificat criteriul Akaike, considerând numărul parametrilor efectivi Π_{ef}, iar Larsen a arătat că, modul de calcul al numărului parametrilor efectivi depinde de doi termeni, estimaţi statistic şi a detaliat expresia obţinută de Moody [L33].

❒ ***Limitarea numărului de parametri ai reţelei*** **(II)**

O regulă empirică pentru obţinerea unei generalizări bune, indică utilizarea structurii neurale cu dimensiuni minime, care poate să înveţe datele din setul de învăţare. Dimensiunile mari ale reţelei presupun existenţa unui număr mare de parametri, care la rândul său permite prea multe generalizări pe baza setului de date de învăţare, dintre care generalizarea corectă este aleasă cu o probabilitate mică [G15].

Intr-o reţea neurală "feedforward", limitarea numărului de parametri se poate face, de exemplu, prin reducerea numărului de noduri ascunse, care conduce la scăderea numărului ponderilor. Există o valoare optimă H_{optim} (fig. 3.15), care se determină prin metode de proiectare (cap. 4), ce reprezintă soluţii la problema generală a selectării modelului neural.

Ideea alegerii dimensiunilor minime ale structurii neurale este în concordanţă cu principiul parsimoniei (par. 4.1).

❒ ***Alegerea numărului de patternuri de învăţare*** **(III)**

Fenomenul de supraînvăţare, urmat de generalizare slabă, poate apare şi dacă numărul parametrilor ce trebuie determinaţi (ponderi, praguri) este prea mare, comparativ cu numărul patternurilor de învăţare. Reţelei neurale i se

Scanned with OKEN Scanner

impun prin setul de învăţare prea puţine constrângeri, în timp ce numărul gradelor sale de libertate (parametri) este ridicat. In cadrul teoriei formale a învăţării, a fost enunţată, alături de problema dimensionării structurii neurale, şi cea a determinării volumului de date de învăţare necesar [K48].

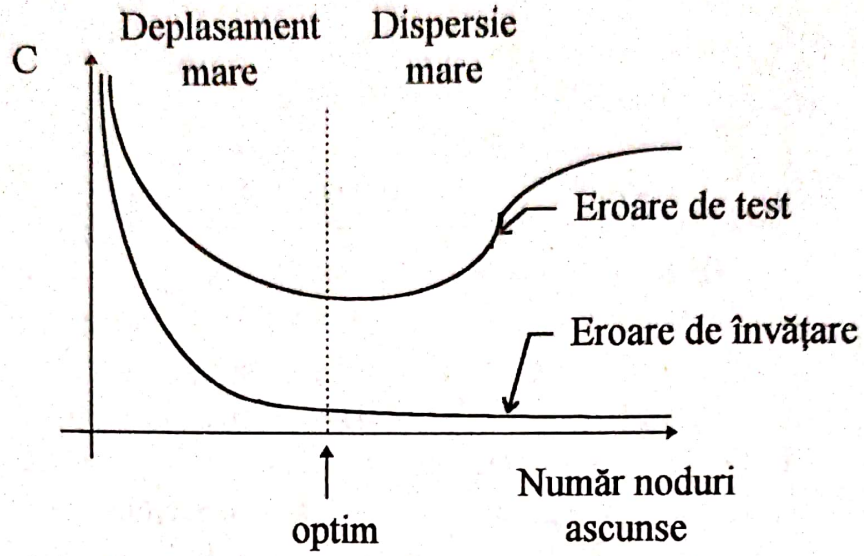


Fig. 3.15 Eroarea de învăţare şi eroarea de test, în funcţie de numărul nodurilor ascunse

3.2.4 Reducerea complexităţii modelului. Regularizare

Selectarea modelului neural cu complexitate (număr de parametri) minimă este o problemă abordată de teoria formală a învăţării [K48]. Argumentele în favoarea rezolvării sale sunt: creşterea capacităţii de generalizare a reţelei, creşterea vitezei de funcţionare a structurii neurale şi scăderea costului implementării.

Metodele de proiectare a structurilor "feedforward" [D205], urmăresc selectarea modelului cu număr minim de parametri, capabil să înveţe setul de date dorit. Regularizarea este o metodă generală de reducere a dimensiunii setului parametrilor. Din perspectiva învăţării ca proces de aproximare, regularizarea reprezintă o tehnică de transformare a problemei învăţării pe baza exemplelor, din "ill posed" (în sensul că informaţia conţinută de date nu este suficientă pentru reconstruirea unică a funcţiei în regiunile unde datele nu sunt disponibile), în problemă "well posed" [L38].

Metoda regularizării a fost propusă în anul 1963 de matematicianul Tihonov. Ideea de bază constă în definirea unui criteriu, pentru a selecta o soluţie aproximativă (funcţie intrare – ieşire) dintr-un set de soluţii admisibile

Scanned with OKEN Scanner

sau, altfel spus, de a face un compromis între fidelitatea faţă de date şi fidelitatea faţă de informaţia apriori despre soluţie. Aceasta se realizează prin includerea în funcţia cost a unui termen suplimentar, ce conţine constrângeri legate de relaţia intrare – ieşire (soluţia problemei de aproximare).

Se notează cu $F_{\mathbf{w}}(\boldsymbol{x})$ (sau $F_{\mathbf{w}}$) funcţia $F(\mathbf{w};\ \mathbf{x})$ implementată de reţeaua neurală, unde $\mathbf{w}$ sunt parametrii acesteia, iar $\boldsymbol{x}$ este vectorul de intrare. Conform teoriei regularizării, funcţia $F_{\mathbf{w}}$ se determină prin minimizarea unei funcţii cost (eroare) regularizată dată de relaţia (3.60), care se poate scrie ca (3.61), deoarece scopul metodei este determinarea funcţiei $F_{\mathbf{w}}$:

$$C(\mathbf{w};\ \mathbf{x};\ \lambda,\ \chi) = \lambda\, C_{\hat{i}nv}(\mathbf{w};\ \mathbf{x}) + \chi\, C_{complexitate}(\mathbf{w};\ \mathbf{x}) \tag{3.60}$$

$$C(F_{\mathbf{w}},\ \lambda,\ \chi) = \lambda\, C_{\hat{i}nv}(F_{\mathbf{w}}) + \chi\, C_{complexitate}(F_{\mathbf{w}}) \tag{3.61}$$

Fără a restrânge generalitatea, se consideră $\lambda = 1$ şi se va utiliza notaţia $C_{\hat{i}nv}$ pentru termenul standard şi notaţiile C_r, $C_{complexitate}$ pentru termenul de regularizare. Primul poate fi, de exemplu, norma L_2 a erorii obţinute la ieşirea reţelei (3.62). Am presupus că intrarea este multidimensională ($M > 1$ intrări), şi există o singură ieşire ($N = 1$) a reţelei.

$$C_{\hat{i}nv}(F_{\mathbf{w}}) = \frac{1}{2}\sum_{\xi=1}^{P}\left[d(\xi) - y(\xi)\right]^2 = \frac{1}{2}\sum_{\xi=1}^{P}\left[d(\xi) - F(\mathbf{w};\ \mathbf{x}(\xi))\right]^2 =$$

$$= \frac{1}{2}\sum_{\xi=1}^{P}\left[d(\xi) - F_{\mathbf{w}}(\mathbf{x}(\xi))\right]^2 \tag{3.62}$$

Termenul de regularizare se poate scrie ca (3.63), unde Φ este un operator liniar (pseudo) diferenţial (cu argumentul inclus în parantezele drepte), care conţine informaţii apriori legate de forma soluţiei (funcţia $F_{\mathbf{w}}$). De aceea, alegerea operatorului Φ este dependentă de problemă. Acesta se numeşte stabilizator, în sensul că determină continuitatea funcţiei $F_{\mathbf{w}}$.

$$C_r(F_{\mathbf{w}}) = \frac{1}{2}\left\|\Phi\left[F(\mathbf{w};\ \mathbf{x})\right]\right\|^2 = \frac{1}{2}\left\|\Phi\left[F_{\mathbf{w}}(\mathbf{x})\right]\right\|^2 \tag{3.63}$$

Pe baza relaţiilor (3.62) – (3.63), funcţia cost ce trebuie minimizată se poate scrie:

$$C(F_{\mathbf{w}},\ \chi) = \frac{1}{2}\sum_{\xi=1}^{P}\left[d(\xi) - F_{\mathbf{w}}(\mathbf{x}(\xi))\right]^2 + \frac{1}{2}\chi\left\|\Phi\left[F_{\mathbf{w}}(\mathbf{x})\right]\right\|^2 \tag{3.64}$$

Scanned with OKEN Scanner

Parametrul de regularizare χ poate fi considerat ca un indicator al modului în care datele de învăţare specifică soluţia $F_w(x)$. Dacă $\chi \to 0$, atunci nu există constrângeri, iar soluţia $F_w(x)$ este complet determinată de setul de date de învăţare; dacă $\chi \to \infty$, atunci constrângerile apriori asupra soluţiei sunt suficiente pentru a o preciza, altfel spus datele de învăţare nu sunt de încredere.

In practică, valoarea parametrului χ se află între aceste limite, astfel încât şi datele, şi constrângerile asupra soluţiei, contribuie la determinarea funcţiei $F_w(x)$. Parametrul χ controlează influenţa termenului C_r de penalizare a complexităţii modelului, asupra soluţiei.

3.2.4.1 Principiul regularizării

Principiul regularizării poate fi sintetizat astfel [H41]:

Să se determine funcţia $F_w(x)$ care minimizează funcţia cost $C(F_w)$, definită de relaţia:

$$C(F_w) = C_{inv}(F_w) + \chi\, C_r(F_w) \qquad (3.65)$$

unde $C_{inv}(F_w)$ este termenul de eroare standard, $C_r(F_w)$ este termenul de regularizare, iar χ este parametrul de regularizare.

3.2.4.2 Soluţia ecuaţiei de regularizare

Soluţia $F_w(x)$ a ecuaţiei de regularizare este o superpoziţie liniară de P funcţii Green, notate $G(x; x(\xi))$ [H41, P3]:

$$F_w(x) = \frac{1}{\chi}\sum_{\xi=1}^{P}\left[d(\xi) - F_w(x(\xi))\right] G(x; x(\xi)) = \sum_{\xi=1}^{P} w(\xi)\, G(x; x(\xi)) \qquad (3.66)$$

care prin definiţie, satisfac ecuaţia (3.67) în fiecare punct, mai puţin în $x(\xi)$:

$$\Phi^* \Phi\, [G(x; x(\xi))] = 0 \qquad (3.67)$$

S-a notat cu Φ^* adjunctul operatorului Φ, cu $x(\xi)$ centrii dezvoltării, iar $w(\xi)$ sunt coeficienţii acesteia. Ca şi anterior, argumentul operatorului $\Phi^* \Phi$ este inclus în parantezele drepte. Soluţia problemei de regularizare se află în subspaţiul P – dimensional al spaţiului funcţiilor, iar baza subspaţiului o

constituie setul funcţiilor Green $\{G(x; x(\xi)\}$ centrate în $x(\xi)$, cu $1 \le \xi \le P$. Numărul funcţiilor Green este egal cu numărul patternurilor de învăţare, iar forma funcţiei depinde de operatorul de stabilizare Φ. In punctul $x(\mu)$, $1 \le \mu \le P$, funcţia (3.66) devine (3.68). Dacă se utilizează notaţiile (3.69) – (3.72) (în care $x(\xi) \overset{not.}{=} x_\xi$, $w(\xi) \overset{not.}{=} w_\xi$), ecuaţiile (3.66) şi (3.68) se pot scrie matricial ca (3.73) şi (3.74) [H41, ş.a.].

$$F_w(x(\mu)) = \sum_{\xi=1}^{P} w(\xi)\, G(x(\mu); x(\xi)) \tag{3.68}$$

$$\mathbf{F_w} = [F_w(x_1), F_w(x_2), \ldots, F_w(x_P)]^T \tag{3.69}$$

$$\mathbf{d} = [d_1, d_2, \ldots d_P]^T \tag{3.70}$$

$$\mathbf{G} = \begin{bmatrix} G(x_1; x_1) & G(x_1; x_2) & \ldots & G(x_1; x_P) \\ G(x_2; x_1) & G(x_2; x_2) & \ldots & G(x_2; x_P) \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ G(x_P; x_1) & G(x_P; x_2) & \ldots & G(x_P; x_P) \end{bmatrix} \tag{3.71}$$

$$\mathbf{w} = [w_1, w_2, \ldots, w_P]^T \tag{3.72}$$

$$\mathbf{w} = \frac{1}{\chi}(\mathbf{d} - \mathbf{F_w}) \tag{3.73}$$

$$\mathbf{F_w} = \mathbf{G}\,\mathbf{w} \tag{3.74}$$

Rezultă relaţia (3.75), din care se obţine (3.76). S-a notat cu I matricea unitate $P \times P$, iar G este matricea Green (matrice simetrică: $G^T = G$).

$$(\mathbf{G} + \chi\, \mathbf{I})\, \mathbf{w} = \mathbf{d} \tag{3.75}$$

Scanned with OKEN Scanner

$$\mathbf{w} = (\mathbf{G} + \chi\, \mathbf{I})^{-1}\, \mathbf{d} \qquad (3.76)$$

Intrucât calculul coeficienţilor **w** ai dezvoltării necesită inversarea unei matrici de dimensiuni P x P (rel. 3.76), iar timpul de calcul creşte polinomial (de ordinul P^3) este necesară reducerea complexităţii problemei, prin determinarea unei soluţii aproximative a problemei de regularizare. Se consideră, conform metodei Galerkin, o soluţie aproximativă $F^*(\mathbf{w}; \mathbf{x})$, pentru care se rescrie funcţia cost (3.64) şi se arată că, termenul de regularizare este dat de relaţia (3.77), unde $\boldsymbol{G}_0$ este o matrice simetrică de dimensiuni Q x Q, cu $Q \leq P$. Vom nota în continuare cu $\boldsymbol{R} = \boldsymbol{G}_0$ (***R*** se numeşte matrice de regularizare) [H41, ş.a.].

$$\left\| \Phi\left[F^*\right] \right\|^2 = \mathbf{w}^T\, \mathbf{G}_0\, \mathbf{w} \qquad (3.77)$$

Rezultă că, pentru $\chi \to 0$, vectorul de ponderi tinde spre soluţia (3.78), unde $\boldsymbol{G}^+$ este matricea pseudoinversă Moore – Penrose[10] a matricii $\boldsymbol{G}$ (3.79) [P3].

$$\mathbf{w} = \mathbf{G}^+\, \mathbf{d} \qquad (3.78)$$

$$\mathbf{G}^+ = (\mathbf{G}^T\, \mathbf{G})^{-1}\, \mathbf{G}^T \qquad (3.79)$$

3.2.4.3 Regularizare cu scăderea ponderilor

Dacă matricea de regularizare este $\boldsymbol{R} = \chi\, \boldsymbol{I}$, cu $\boldsymbol{I}$ matricea unitate, atunci metoda se numeşte "regularizare cu scăderea ponderilor" ("weight decay") [L33], pentru că termenul de penalizare forţează, în funcţie de forma sa specifică, anumite ponderi să tindă spre valori mici. Intrucât scăderea ponderilor are loc în timpul învăţării, rezultă că reducerea structurii este o consecinţă firească a regularizării.

Scăderea ponderilor se poate interpreta din punct de vedere al teoriei Bayes, în sensul că termenul de penalizare corespunde unei ipoteze apriori asupra distribuţiei parametrilor independenţi ai modelului. Alegerea acestuia corespunde preferinţei unui anumit tip de model.

Expresia (3.61) devine, prin aplicarea logaritmului negativ şi ignorării termenilor constanţi, relaţia (3.80), unde $p(\mathbf{w} \mid \Delta)$ este densitatea de probabilitate

---

[10] Dacă se presupune că valorile proprii ale matricii $G(\mathbf{w}^*)$ respectă relaţia: $\lambda_1 \geq .. \geq \lambda_n > 0$ şi $\lambda_i = 0$, $\forall i \in [n+1, I]$, iar vectorii proprii asociaţi sunt plasaţi ca vectori coloană în matricea Q, atunci matricea pseudoinversă este [L33]:

$$\mathbf{G}^+(\mathbf{w}^*) = \mathbf{Q}\ \mathrm{diag}\left[\lambda_1^{-1}, \ldots, \lambda_n^{-1}, 0, \ldots, 0\right]\ \mathbf{Q}^T$$

Scanned with OKEN Scanner

în spaţiul ponderilor, $p(\Delta \mid \mathbf{w})$ este probabilitatea datelor de învăţare ($\Delta = \{(x(\xi); d(\xi), 1 \le \xi \le P\}$), iar $p(\mathbf{w})$ este probabilitatea apriori a ponderilor.

$$p(\mathbf{w} \mid \Delta) \propto p(\Delta \mid \mathbf{w})\, p(\mathbf{w}) \tag{3.80}$$

Pentru $p(\Delta \mid \mathbf{w})$ din relaţia (3.81) şi $p(\mathbf{w})$ din (3.82), unde $Z_{\hat{i}nv} = Z_{\hat{i}nv}(\lambda)$ şi $Z_r = Z_r(\chi)$ sunt constante de normalizare, rezultă că procesul de minimizare a expresiei (3.83) este echivalent determinării densităţii maxime de probabilitate posteriori.

$$p(\Delta \mid \mathbf{w}) = Z_{\hat{i}nv}^{-1} \exp\left(-\lambda C_{\hat{i}nv}\right) \tag{3.81}$$

$$p(\mathbf{w}) = Z_r^{-1} \exp\left(-\chi C_r\right) \tag{3.82}$$

$$C(\mathbf{w}) = -\log p(\mathbf{w} \mid \Delta) + \text{constantă} \tag{3.83}$$

Ipoteza apriori asupra distribuţiei parametrilor independenţi ai modelului, reflectată în forma termenului de penalizare din funcţia cost, poate fi gaussiană, Laplace, Cauchy, etc. [W19]. Dacă componentele vectorilor de ponderi sunt $\{w_q \mid 1 \le q \le \Pi\}$, cu Π numărul maxim de ponderi, atunci termenul de regularizare al funcţiei cost are una din formele prezentate în tab. 3.6. Odată aleasă distribuţia apriori, iniţializarea ponderilor trebuie realizată corespunzător acesteia.

Se poate arăta că, distribuţia de tip Laplace este mai indicată pentru ponderile conexiunilor spre şi de la nodurile ascunse ale reţelei "feedforward", decât distribuţia Gauss, în sensul unui efect direct de reducere a structurii neurale (pe baza observaţiei că, derivata termenului $C_{\hat{i}nv}$ al funcţiei cost este în cele două cazuri diferită). Pe de altă parte, metodele de penalizare care adaugă termeni pătratici de forma $\sum_q w_q^2$ în funcţia cost au dezavantajul că, favorizează vectorii de ponderi cu multe componente de valoare mică, în defavoarea celor care au o singură componentă de valoare mare [W19, D205].

3. 3 REZUMATUL CAPITOLULUI

Am prezentat bazele teoretice ale abordării modelelor neurale artificiale, insistând asupra caracteristicilor generale ale acestora şi apoi, asupra reţelelor "feedforward", care vor constitui obiectul studiului în continuare.

Scanned with OKEN Scanner

Tab. 3.6 Distribuţii apriori posibile ale ponderilor

Distribuţie apriori a ponderilor	Termenul de penalizare C_r al funcţiei cost
Gauss	$\frac{1}{2}\sum_{q=1}^{\Pi} w_q^2$
Laplace	$\frac{1}{2}\sum_{q=1}^{\Pi} \lvert w_q \rvert$
Cauchy	$\frac{1}{\lambda}\sum_{q=1}^{\Pi} \log\left(1+\lambda^2 w_q^2\right)$ cu $(1/\lambda)$ valoare mediană

Prima parte a capitolului prezintă caracteristicile modelelor neurale (cele ale micro – structurii: modele de perceptron simplu cu şi fără reacţie, funcţii de activare; topologia reţelei, la nivel de mezo– şi macro – structură "feedforward" şi "feedback"; modelul datelor şi criteriul de cost; legea de învăţare; condiţii de stabilitate), ansamblul acestora definind reţeaua neurală artificială.

Cea de-a doua parte a capitolului se referă la aspecte teoretice legate de reţelele neurale "feedforward": învăţarea (ca proces de aproximare şi ca proces de optimizare), algoritmi de învăţare. Sunt precizate apoi, definiţia generalizării în RNA "feedforward", modalităţi de evaluare a acesteia, precum şi variante de îmbunătăţire a capacităţii de generalizare. In finalul capitolului, este prezentată regularizarea, ca metodă generală de reducere a complexităţii modelului neural "feedforward". Enunţul principiului este urmat de soluţia ecuaţiei de regularizare şi de varianta de regularizare cu scăderea ponderilor.

Capitolul concentrează noţiunile teoretice de bază, necesare abordării în continuare, a proiectării RNA.

Scanned with OKEN Scanner

4 METODE DE PROIECTARE A STRUCTURILOR NEURALE "FEEDFORWARD"

4.1 ALEGEREA MODELULUI NEURAL. CRITERII DE OPTIMALITATE

Pentru realizarea unei aplicaţii cu RNA, este necesară alegerea corectă a modelului neural şi apoi dimensionarea sa. Reţeaua rezultată în urma acestor etape de proiectare este în cazul ideal optimală, dar dependentă de aplicaţie.

In continuare prezentăm criterii de evaluare a modelului, argumente în favoarea reducerii complexităţii acestuia şi criterii de optimalitate.

4.1.1 Criterii de evaluare a modelului

Harvey a sugerat un set de criterii (❶–❻) pentru evaluarea modelelor, care se pot aplica şi modelelor neurale artificiale. Respectarea lor defineşte un model "bun" [C38]. Criteriilor ❶–❻ li se adaugă cele formulate de Stone (❼, ❽) [J3].

❶ Parsimonie

Principiul parsimoniei ("parsimony principle") a fost enunţat de filozoful medieval William of Ockham (? 1300 – 1349). Ideea generală afirmată de acesta este aceea că[11].

Nu trebuie sa existe mai multe entitati decât este necesar.

Deşi nu a fost demonstrată în cazul general, ideea constituie un principiu de bază al ştiinţei moderne şi indică necesitatea simplificării teoriilor ştiinţifice. In cazul particular al reţelelor neurale, se poate formula astfel: "fiind date două reţele neurale, şi acelaşi set de date de învăţare pentru problema dată, reţeaua cu numărul minim de parametri va 'ghici' (= generaliza) mai bine rezultatul în viitor, pe baza unor exemple ale problemei". Altfel spus, RNA respectă principiul "Lama / Briciul lui Ockham" ("Ockham's razor") dacă numărul parametrilor săi (indicator al complexităţii) este redus [A22, K48].

[11] Deşi a constituit bază de inspiraţie pentru teorii ştiinţifice ulterioare, ideea nu a fost exprimată efectiv, în scrierile lui Ockham. Acesta a anticipat însă în studiile sale multe dintre concepţiile filozofice de mai târziu, incluzând teoria cauzalităţii a lui Hume, teoria relaţiilor a lui Leibniz, etc. [S208].

Scanned with OKEN Scanner

❷ Coerenţa datelor

Modelul neural trebuie să poată învăţa setul de date, conform criteriului de cost ales.

❸ Consistenţa cu toate cunoştinţele apriori

Acest criteriu este relevant numai dacă s-au utilizat cunoştinţe apriori pentru construirea modelului.

❹ Admisibilitatea datelor

Modelul trebuie să respecte constrângerile rezonabile impuse.

❺ Stabilitate structurală

Funcţionarea modelului trebuie să fie corectă, chiar pentru condiţii iniţiale diferite.

❻ "Excelenţă" ("encompassing")

Rezultatele obţinute cu modelul ales (exemplu: modelul neural) trebuie să fie mai bune decât cele obţinute folosind modele alternative.

❼ Flexibilitate

❽ Dimensionalitate

Dimensiunea setului de date necesar obţinerii unui estimat suficient de bun al parametrilor modelului (ponderi, praguri) creşte rapid cu numărul variabilelor de intrare ale acestuia. Intrucât nu poate fi oricât de mare, limitarea sa reprezintă respectarea criteriului de dimensionalitate.

❾ Interpretabilitate

Interpretabilitatea este capacitatea modelului de a furniza informaţii despre structura cu care se rezolvă problema dată.

4.1.2 Argumente în favoarea reducerii complexităţii modelului

Există multe argumente în favoarea reducerii complexităţii modelului. Fără a restrânge generalitatea, presupunem că modelul ce respectă criteriile ❶–❾ este "feedforward" din punct de vedere structural.

❒ ***Invăţarea este NP – completă***

Blum şi Rivest [N6] au arătat, pentru o structură "feedforward" cu numai trei straturi, M intrări, două noduri ascunse şi un singur nod de ieşire, învăţarea

în rețea ca proces de optimizare este NP – completă. In consecință, chiar atunci când dimensiunea rețelei este determinată mai ales de dimensiunea stratului de intrare, timpul de învățare este mai mare decât o funcție polinomială de suma dintre dimensiunea rețelei și cea a problemei studiate. Este important faptul că, rezultatul obținut de Blum și Rivest este independent de algoritmul utilizat pentru determinarea ponderilor.

- ***Generalizarea crește prin reducerea complexității modelului***

In reprezentarea grafică a erorii în timpul învățării și erorii de test, în funcție de numărul nodurilor ascunse ale rețelei (fig. 3.15) se poate observa apariția fenomenului de supraînvățare, dacă numărul parametrilor rețelei depășește o valoare de prag. Am menționat că, fenomenul amintit poate fi evitat printre altele, prin limitarea numărului de parametri ai RNA (par. 3.2.3), în acest caz determinat de alegerea diferită a numărului nodurilor ascunse. O regulă empirică pentru obținerea unei generalizări bune este utilizarea celei mai mici rețele care este capabilă să învețe setul de date de învățare.

- ***Creșterea vitezei modelului în etapa de test***

Numărul operațiilor de adunare și înmulțire, specifice majorității algoritmilor RNA, depinde de numărul parametrilor structurii. Reducerea acestuia determină un volum de calcul mai redus în etapa de test, deci viteză mai mare de funcționare și posibilitatea de lucru în timp real.

- ***Scăderea costului implementării***

Reducerea numărului parametrilor ce trebuie stocați determină scăderea costului implementărilor RNA, atât hardware cât și software. Argumentelor în favoarea reducerii complexității modelului neural li se adaugă consonanța cu principiul parsimoniei. In acest caz, devine echivalentă termenului "simplificare" din enunțul acestuia.

4.1.3 Avantajele și dezavantajele structurilor neurale "feedforward" de dimensiune redusă

In structurile neurale "feedforward" de dimensiune redusă se evită apariția fenomenului de supraînvățare. De asemenea, scade probabilitatea de estimare incorectă a ponderilor în etapa de învățare. Generalizarea este mai bună, iar viteza în etapa de test este mai ridicată. Pe de altă parte, implementarea are un

Scanned with OKEN Scanner

cost mai scăzut, iar extragerea regulilor dintr-o RNA care a învăţat, în scopul construirii unui sistem expert, se simplifică.

Principalul dezavantaj al structurilor neurale de dimensiune redusă este scăderea toleranţei la defecte, prin micşorarea redundanţei.

4.1.4 Criterii de optimalitate

Proiectarea reţelelor neurale "feedforward", cu respectarea criteriilor prezentate în par. 4.1.1 şi dimensionare minimă conduce la arhitecturi care în cazul ideal, sunt optime. Obţinerea unor performanţe cât mai bune prin optimizarea structurii neurale este o problemă fundamentală de proiectare [L33].

Criteriul de evaluare a optimalităţii poate fi obţinerea minimului unuia dintre indicatorii: ❶ eroare medie de generalizare, ❷ timp de învăţare, ❸ dimensiunea structurii (evaluată prin numărul total de perceptroni simpli).

Definirea arhitecturii neurale optimale ca reţeaua cu dimensiuni minime, implică scăderea timpului de învăţare, precum şi generalizare mai bună. Pe de altă parte, arhitectura neurală optimală poate fi definită ca reţeaua care minimizează eroarea medie de generalizare Γ care este, conform relaţiei (3.52) suma a trei componente. Minimizarea sa înseamnă realizarea unui compromis între acestea.

Criteriile de optimalitate amintite nu sunt independente. Relaţia dintre ele este prezentată în fig. 4.1 [D208]. In consecinţă, ne vom referi la reducerea dimensiunilor structurii neurale, având în vedere implicaţiile acesteia asupra capacităţii de învăţare şi erorii medii de generalizare.

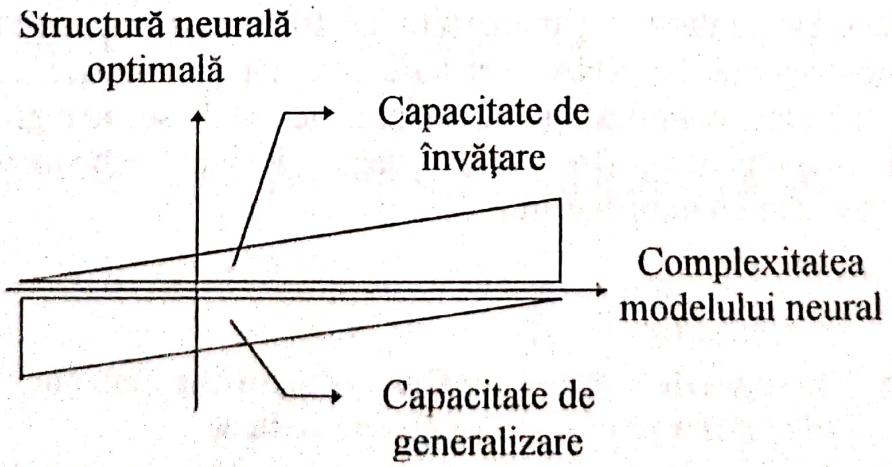


Fig. 4.1 Relaţia dintre învăţare, generalizare şi complexitatea modelului neural

4.2 OBIECTIVUL ŞI ETAPELE PROIECTĂRII STRUCTURILOR NEURALE "FEEDFORWARD"

Deşi esenţială în aplicarea practică a RNA, proiectarea acestora este dificilă şi presupune rezolvarea a două probleme distincte: determinarea dimensiunilor structurii neurale şi estimarea valorilor ponderilor. Cea din urmă se realizează prin procesul de învăţare. In continuare ne vom referi la alegerea dimensiunilor structurii, care are o importanţă atât teoretică, cât şi practică. In prezent nu există modalităţi formale de determinare a acestora, în funcţie de complexitatea problemei.

Teoriile de generalizare curente estimează eroarea de test pe baza erorii de învăţare şi capacităţii modelului. Având în vedere că, un număr prea mare de parametri permite prea multe generalizări pe baza setului de învăţare, dintre care cea corectă este aleasă cu o probabilitate mică [G15], rezultă regula de proiectare (4.1). Dimensionarea reţelei neurale "feedforward" conform acesteia, se realizează specificând numărul de straturi, precum şi cel al perceptronilor simpli de pe fiecare strat.

Minimizează dimensiunile reţelei neurale cu constrângerile că, aceasta este capabilă să înveţe setul datelor de învăţare	(4.1)

4.2.1 Alegerea numărului de straturi

Conform teoremelor de reprezentare (par. 3.2.2.1), o reţea neurală "feedforward" perceptron multistrat cu maximum două straturi active (un strat ascuns şi un strat de ieşire) este capabilă să aproximeze majoritatea funcţiilor neliniare întâlnite în aplicaţiile practice. De asemenea, un perceptron multistrat cu trei straturi active (două straturi ascunse şi un strat de ieşire) este capabil să aproximeze orice funcţie neliniară. In consecinţă, maximum două straturi ascunse sunt suficiente.

In aplicaţii practice este posibil ca numărul perceptronilor simpli de pe fiecare din cele unul / două straturi ascunse să fie mult prea mare pentru un sistem de dimensiuni rezonabile. De aceea, utilizarea RNA "feedforward" cu mai multe straturi ascunse, compuse dintr-un număr redus de noduri, oferă uneori o soluţie problemei date şi nu contrazice teoremele de reprezentare (conform cărora există teoretic şi o structură cu numai unul / două straturi ascunse, care să o poată rezolva).

Scanned with OKEN Scanner

4.2.2 Alegerea numărului de perceptroni simpli din fiecare strat

Dimensiunea stratului de intrare al reţelei neurale "feedforward" este egală cu cea a patternurilor de învăţare.

Numărul perceptronilor simpli din stratul de ieşire se determină în funcţie de problema dată. De exemplu, dacă aceasta este clasificare, atunci numărul nodurilor de ieşire este egal cu numărul claselor în care sunt distribuite patternurile de intrare.

Dimensionarea stratului ascuns al reţelei neurale "feedforward" este o problemă dificilă, care influenţează performanţele obţinute. Dacă numărul nodurilor ascunse este prea mic, atunci sunt modificate prea puţine ponderi şi modelul neural nu va putea realiza corespunzător aproximarea dorită. Valoarea funcţiei cost $C_{\text{înv}}$ va fi prea mare pentru a fi acceptabilă.

Dacă numărul nodurilor ascunse este prea mare, atunci numărul parametrilor (ponderi) este ridicat, iar eroarea de învăţare $C_{\text{înv}}$ este mică. Pe de altă parte, setul de date de învăţare este finit ($P < \infty$), deci dispersia valorilor estimate ale parametrilor modelului creşte, pe măsură ce numărul acestora se măreşte. Valorile ponderilor estimate imprecis conduc, mai departe, la o valoare ridicată a erorii de generalizare C_g. De aceea, valoarea optimă a numărului de noduri ascunse este cea care minimizează eroarea globală C dată de relaţia (3.51) [H30].

In tab. 4.1 este prezentată o clasificare a metodelor de dimensionare a stratului ascuns în structurile neurale "feedforward". Metoda este considerată directă dacă rezultatul său este numărul nodurilor ascunse, respectiv indirectă dacă determină numărul parametrilor (ponderilor) structurii (numărul nodurilor ascunse se obţine indirect).

Tab. 4.1 Dimensionarea stratului ascuns în structurile neurale "feedforward"

Metode de dimensionare a stratului ascuns		Tipul metodelor
Metode empirice		Directe
Metode bazate pe criterii statistice		Indirecte
Metode ontogenice	Constructive	Directe
	Distructive	Directe
	Mixte	Directe
	Bazate pe algoritmi genetici	Directe

Scanned with OKEN Scanner

4.2.3 Alegerea numărului de conexiuni

Dacă structura neurală este total interconectată (fiecare perceptron simplu i din stratul L este conectat cu fiecare perceptron simplu din stratul L+1), atunci numărul conexiunilor existente între toate nodurile rețelei neurale "feedforward" este egal cu cel al ponderilor şi se determină prin calcul simplu, presupunând cunoscute dimensiunile fiecărui strat.

4.2.4 Proiectarea rețelelor neurale ontogenice

Clasa de rețele neurale care îşi modifică structura, astfel încât să nu mai fie necesară stabilirea acesteia înaintea învățării, este numită de Fiesler clasa "rețelelor ontogenice"[12] sau "ontogenetice", proiectate prin metodele cu acelaşi nume. Modificările de topologie posibile sunt: adăugarea / ştergerea straturilor, perceptronilor simpli sau conexiunilor. Modificarea numărului de straturi implică de obicei modificarea numărului de noduri, iar modificarea acestora implică de obicei modificarea numărului conexiunilor [F28].

Dacă structura inițială este mică şi se adaugă perceptroni simpli / straturi atunci metoda este "constructivă" [H19]. Dacă RNA inițială are dimensiuni mari şi se elimină treptat conexiuni / noduri / straturi, atunci metoda se numeşte "distructivă". Există posibilitatea combinării celor două tipuri de metode în variante hibride. O altă posibilitate este utilizarea algoritmilor genetici atât pentru creşterea, cât şi pentru descreşterea structurii (o clasă de metode cu trăsături specifice).

In continuare ne vom referi la metode de proiectare a RNA "feedforward", având în vedere dimensionarea stratului ascuns (tab. 4.1) şi utilizând notațiile introduse în par. 3.2.1.

4.3 METODE EMPIRICE PENTRU PROIECTAREA STRUCTURILOR NEURALE "FEEDFORWARD"

O soluție simplă a problemei de proiectare (4.1) este învățarea mai multor rețele pe un set de date, până criteriul de cost este îndeplinit şi apoi alegerea structurii cu dimensiunea minimă. Metoda este însă consumatoare de timp, de aceea s-au propus variante empirice care nu aibă acest dezavantaj.

[12] Ontogenic sau ontogenetic (DEX) = care aparține ontogenezei;
Ontogeneza (DEX) = Dezvoltare individuală a organismelor vegetale şi animale, care cuprinde toate transformările organismului de la stadiul de embrion până la sfârşitul existenței lui.

Scanned with OKEN Scanner

4.3.1 Metode ad hoc

❒ ***Relația Chauvin* (I)**

Yves Chauvin a studiat experimental efectul numărului diferit de noduri ascunse asupra performanțelor obținute în problema de clasificare SAU–EXCLUSIV (XOR). Relația empirică obținută este:

$$P = 280 - 33 \log_2 H \tag{4.2}$$

cu H numărul nodurilor ascunse, P numărul total al patternurilor [R3]. Relația se poate utiliza pentru determinarea valorii H, dacă P este cunoscut.

❒ ***Relația Kudrycki* (II)**

Kudrycki a determinat empiric rata optimă de 3:1 a primului strat ascuns față de cel de-al doilea, chiar și pentru vectori de intrare de dimensiuni mari [M17]:

$$H_1 = 3 H_2 \tag{4.3}$$

❒ ***Relația Hecht – Nielsen* (III)**

Interpretând teorema Kolmogorov, Hecht – Nielsen menționa că, orice funcție continuă poate fi aproximată de rețeaua cu un număr de noduri ascunse în stratul ascuns [H7]:

$$H = 2 M + 1 \tag{4.4}$$

unde s-a notat cu M numărul perceptronilor simpli de intrare.

❒ ***Relația Baum – Haussler – Widrow* (IV)**

Baum și Haussler [B500] au arătat că, o rețea "feedforward" cu M noduri liniare (perceptroni simpli având funcții de activare liniare) și Π ponderi, necesită un număr de patternuri de învățare egal cu:

$$P > O\left(\frac{\Pi}{\varepsilon} \log\left(\frac{M}{\varepsilon}\right)\right) \tag{4.5}$$

pentru a învăţa cu o eroare mai mică de $\varepsilon / 2$, unde $0 < \varepsilon < 1/8$. Relaţia (4.5) se aproximează mai departe, prin (4.6) [N6], care poate fi utilizată pentru determinarea numărului parametrilor Π, deci indirect şi a valorii H. Intrucât într-o reţea cu un singur strat ascuns, total interconectată, numărul parametrilor se poate calcula cu relaţia (4.7), atunci folosind (4.6), rezultă că alegerea valorii H se poate face respecând (4.8).

$$P > \frac{\Pi}{\varepsilon} \tag{4.6}$$

$$\Pi = H(M + 1) + (H + 1)N = (M + N + 1)\, H + N \tag{4.7}$$

$$H \leq \frac{\varepsilon P}{M + N} \tag{4.8}$$

Relaţia (4.6) a fost obţinută şi de Widrow, în mod independent, dar inegalitatea nu mai este strictă.

4.3.2 Metode bazate pe stabilirea tipului şi numărului zonelor de decizie realizate de reţeaua neurală în spaţiul multidimensional

❒ ***Metoda* (I)**

In probleme de clasificare, în care valoarea ieşirii face parte dintr-un set finit de numere diferite (clase), se poate realiza legătura dintre numărul de perceptroni simpli ascunşi şi cel al regiunilor separabile liniar (se presupune o funcţie de activare a neuronilor de tip treaptă unipolară). Acest tip de analiză are drept scop găsirea celui mai mic număr de noduri ascunse, astfel încât clasificarea să fie posibilă în cazul unei probleme particulare, în care numărul regiunilor de decizie separabile liniar este cunoscut.

Dacă M, H, N sunt dimensiunile straturilor reţelei neurale "feedforward" şi P este numărul minim de patternuri din setul de învăţare, se presupune că spaţiul N – dimensional este împărţit în zone disjuncte, fiecare aparţinând unei clase C_j, cu $j \leq N$. Intuitiv, numărul de reuniuni N indică dimensiunea minimă a setului P de patternuri de învăţare, deci este necesară îndeplinirea condiţiei $P \geq N$. Intre M, H, N există o relaţie care permite calcularea unuia dintre aceştia, când se cunosc ceilalţi.

Numărul maxim de regiuni separabile liniar într-un spaţiu de intrare M – dimensional, cu ajutorul a H perceptroni simpli ascunşi, având funcţii de activare de tip treaptă unipolară, este [Z2]:

Scanned with OKEN Scanner

$$\text{NrReg}(H, M) = \arg \max_{H} \sum_{m=1}^{M} \binom{H}{m} \tag{4.9}$$

unde $\binom{H}{m} = 0$ pentru $H < m$. Acest lucru poate fi demonstrat prin inducţie [M501]. Dacă numărul regiunilor separabile liniar $NrReg_{lin}$ este cunoscut, atunci valoarea minimă a numărului de noduri ascunse H, notată cu H_{min}, poate fi calculată cu relaţia (4.10). Cu toate acestea, valoarea $NrReg_{lin}$ nu este cunoscută apriori în majoritatea problemelor reale.

$$H_{min} = \min_{H} \{\text{Nr Reg}(H, M) \geq \text{Nr Reg}_{lin}\} \tag{4.10}$$

Pentru vectori de intrare de dimensiuni mari, comparativ cu numărul nodurilor ascunse, adică $M \geq H$ se poate scrie relaţia:

$$N = \binom{H}{0} + \binom{H}{1} + \binom{H}{2} + \cdots + \binom{H}{H} = 2^H \tag{4.11}$$

Relaţiile (4.9) şi (4.11) pot fi inversate, pentru determinarea numărului de perceptroni simpli H necesari pentru a obţine clasificarea datelor M–dimensionale în N clase. Valoarea H este soluţia ecuaţiei (4.12), pentru $H > M$.

Pentru $H \leq M$, valoarea H obţinută din relaţia (4.11) este (4.13).

$$N = 1 + H + \frac{H(H-1)}{2!} + \frac{H(H-1)(H-2)}{3!} + \cdots + \frac{H(H-1)\cdots(H-M+1)}{n!} \tag{4.12}$$

$$H = \log_2 N \tag{4.13}$$

❒ *Metoda Lippmann* (II)

Lippmann a oferit un argument geometric elegant în sprijinul determinării numărului nodurilor ascunse. Acesta se referă la cazul funcţiilor de activare de tip prag, dar se poate extinde pentru funcţii sigmoidale. Reţeaua are valori de ieşire binare.

Lippmann precizează că, fiecare nod din primul strat ascuns realizează un plan de decizie (sau hiperplan), care este bisector al spaţiului de intrare M – dimensional (M este numărul perceptronilor simpli de intrare). Fiecare nod din al

Scanned with OKEN Scanner

doilea strat ascuns adaugă unul sau mai multe subspaţii create de aceste plane de decizie, pentru a da naştere unor zone de decizie convexe. Fiecare nod din stratul de ieşire adaugă unul sau mai multe astfel de subspaţii convexe, pentru a crea un subspaţiu de decizie arbitrar [L4].

Numărul de perceptroni simpli din stratul ascuns se poate calcula cu relaţia:

$$H = N\,(M + 1) \tag{4.14}$$

4.4 METODE DE PROIECTARE BAZATE PE CRITERII STATISTICE

4.4.1 Formularea problemei

Obţinerea unor performanţe bune pe setul de test şi reducerea numărului de parametri ai reţelei neurale sunt două cerinţe contradictorii. Compromisul se poate realiza dacă se ia în considerare relaţia dintre complexitatea modelului şi performanţa sa pe setul de date de învăţare [L35, L36]. Complexitatea modelului este evaluată prin numărul parametrilor independenţi necesari construirii unui model neliniar al fenomenului fizic. Acesta este caracterizat prin setul de date intrare – ieşire dorită, utilizate pentru învăţarea reţelei.

Problema de proiectare se poate formula astfel [H39]:

Fiind dat un set de date $\boldsymbol{x} = \{x_i;\ 1 \le i \le M\}$,
se doreşte estimarea modelului $\mathbf{M}$, caracterizat de funcţia
densitate de probabilitate $p(\boldsymbol{x}, \mathbf{w}_\Pi)$, (4.15)
unde s-au notat cu:

$\mathbf{w}_\Pi$	setul de parametri independenţi ai modelului
Π	numărul parametrilor modelului

In ultimii ani a existat o preocupare intensă pentru găsirea criteriilor de selectare a modelului: Akaike, Schwartz, MacKay şi Moody au contribuit, printre alţii, la analiza diferitelor variante propuse în acest scop [G17]. Metodele statistice de selectare a modelului utilizează una dintre variantele următoare:

- Alegerea modelului şi apoi efectuarea unui număr de teste statistice. Dezavantajul este acela că, metodele de acest tip sunt consumatoare de timp, numărul testelor fiind uneori mare.
- Găsirea unui criteriu, care să ia în considerare caracteristicile de supraînvăţare ("overfitting") şi de subînvăţare ("underfitting") ale modelului.

Scanned with OKEN Scanner

4.4.2 Forma generală a criteriilor statistice pentru proiectarea structurilor neurale "feedforward"

Deşi criteriile statistice pentru proiectarea structurilor neurale "feedforward" diferă, fiecare dintre acestea se poate obţine prin particularizarea formei generale:

$$\begin{pmatrix}\text{Complexitatea}\\ \text{modelului}\end{pmatrix} = \begin{pmatrix}\text{Funcţie logaritm din}\\ \text{densitatea de}\\ \text{probabilitate}\end{pmatrix} + \begin{pmatrix}\text{Termen de penalizare a}\\ \text{complexităţii modelului}\end{pmatrix}$$

care se poate scrie ca relaţia (3.60) care, pentru $\lambda = 1$ şi considerând funcţia cost globală, devine:

$$C(\mathbf{w}) = C_{\text{înv}}(\mathbf{w}) + \chi\, C_{\text{complexitate}}(\mathbf{w}) \tag{4.16}$$

Funcţia cost standard $C_{\text{înv}}(\mathbf{w})$, evaluată pe setul de învăţare, depinde atât de modelul neural, cât şi de datele de intrare, şi este o funcţie de estimatul dispersiei erorii asociate reţelei, notat $\hat{\sigma}^2_{\Pi}$, cu Π numărul de parametri independenţi ai modelului.

Diferenţele între criteriile statistice se referă mai ales la forma termenului de penalizare a complexităţii modelului, $C_{\text{complexitate}}(\mathbf{w})$, care depinde numai de model. Alegerea termenilor $C_{\text{înv}}(\mathbf{w})$, $C_{\text{complexitate}}(\mathbf{w})$ şi χ determină caracteristicile criteriului, (tab. 4.2), cu P numărul observaţiilor efectuate.

O altă exprimare a criteriului general (4.15) este considerată relaţia (4.17), unde s-a notat cu $\hat{\mathbf{w}}_{\Pi} = \hat{\mathbf{w}}_{\Pi}(\mathbf{x})$ estimatorul în sensul verosimilităţii maxime ML ("maximum likelihood"). Pentru diferite criterii, termenul $C_{\text{complexitate}}(\Pi)$ are forma din tab. 4.3, în care s-au folosit notaţiile (4.18) şi (4.19).

$$C(\Pi, \mathbf{M}) = -\log\, p(\hat{\mathbf{w}}_{\Pi}, \mathbf{x}) + C_{\text{complexitate}}(\Pi) \tag{4.17}$$

Tab. 4.2 Alegerea mărimilor $C_{\text{înv}}(\mathbf{w})$ şi χ pentru diferite criterii statistice, când $C_{\text{complexitate}}(\mathbf{w}) = \Pi$

Criteriul statistic	$C_{\text{înv}}(\mathbf{w})$	χ
Akaike (AIC)	$P \log\left(\hat{\sigma}^2_{\Pi}\right)$	2
Schwartz		log P
Rissanen (MDL)		log P
Hannan – Quinn		c log (log P), (c = const., $c > 2$)

Scanned with OKEN Scanner

Tab. 4.3 $C_{complexitate}(\Pi)$ pentru criteriile TIC, AIC şi MDL
TIC = criteriul AIC modificat de Takeuchi;
MDL = criteriul "Minimum Description Length"

Criteriul statistic	$C_{complexitate}(\Pi)$
TIC	$-\operatorname{tr} \hat{J}^{-1} \hat{Q}$
AIC	Π
MDL	$(\Pi / 2) \log P$

$$\hat{J} = \left. \frac{\partial^2 \log p(\mathbf{w}_\Pi, \mathbf{x})}{\partial \mathbf{w}_\Pi \, \partial \mathbf{w}_\Pi^T} \right|_{\hat{\mathbf{w}}_\Pi} \tag{4.18}$$

$$\hat{Q} = \left. \left[\frac{\partial \log p(\mathbf{w}_\Pi, \mathbf{x})}{\partial \mathbf{w}_\Pi} \; \frac{\partial \left(\log p(\mathbf{w}_\Pi, \mathbf{x}) \right)}{\partial \mathbf{w}_\Pi^T} \right] \right|_{\hat{\mathbf{w}}_\Pi} \tag{4.19}$$

Se doreşte selectarea modelului (determinarea numărului Π de parametri), care minimizează relaţia (4.16), respectiv (4.17) [H39]. De exemplu, criteriile AIC şi MDL pot fi scrise sub forma:

$$\mathrm{AIC}(\Pi) = -\max_{\mathbf{w}} \left(\log C_{\hat{inv}} \right) + \Pi \tag{4.20}$$

$$\mathrm{MDL}(\Pi, P) = -\max_{\mathbf{w}} \left(\log C_{\hat{inv}} \right) + \frac{\Pi}{2} \log P \tag{4.21}$$

Se observă că, relaţia (4.16) este expresia funcţiei de cost a reţelei regularizate, cu χ parametru de regularizare. Criteriile statistice urmăresc deci, micşorarea complexităţii modelului, utilizând regularizarea. Tehnici de reducere a reţelei pe baza teoriei regularizării sunt prezentate în cap. 4.6.

4.4.3 Criterii statistice pentru proiectarea structurilor neurale "feedforward"

Evaluarea calităţii modelului determinat, având complexitate minimă, se poate face pe baza erorii medii de generalizare medie, a tuturor seturilor de date posibile, având dimensiunea P, cu P numărul patternurilor de învăţare.

Considerăm că, pentru modelul neural "feedforward" dat de relaţia (3.53), se realizează minimizarea funcţiei cost (3.54), unde C_{inv} este (3.55). Valoarea

medie a funcţiei cost se calculează cu expresia (3.57), iar eroarea medie de generalizare este dată de (3.59). Se doreşte găsirea unui estimat al acesteia, notat $\hat{\Gamma}$, conform definiţiei 3.23.

❒ *Criteriul FPE ("Final Prediction Error")* (I)

Criteriul FPE ("Final Prediction Error") propus de Akaike în anul 1969, exprimă relaţia dintre eroarea de test (notată cu FPE pentru acest criteriu) şi cea de învăţare (relaţiile 4.22, 4.23) [A500, J18]. Se aplică pentru estimarea erorii medii de generalizare (test), folosind valoarea erorii de învăţare şi numărul de parametri Π ai modelului. Inainte de aplicarea relaţiei, din valorile datelor trebuie extrasă media acestora.

$$\hat{\Gamma}_{FPE} = E_{seturi\ P}\{ C_{inv}(\mathbf{w})\} \frac{P + \Pi}{P - \Pi} \tag{4.22}$$

$$\hat{\Gamma}_{FPE} = C_{inv}(\hat{\mathbf{w}}) \frac{P + \Pi}{P - \Pi} \tag{4.23}$$

Din relaţia (4.23) se poate determina numărul Π al parametrilor modelului. Criteriul a fost des utilizat, deşi Kashyap [L35], a demonstrat că are tendinţa de a determina supraînvăţarea. De asemenea, relaţia (4.22) indică faptul că, eroarea de învăţare şi cea de generalizare sunt estimatori deplasaţi ai nivelului de zgomot din datele de intrare, deoarece fiecare parametru "a absorbit" în timpul învăţării eroare din patternurile de învăţare.

❒ *Criteriul AIC ("Akaike's Information Theoretic Criterion")* (II)

Criteriul propus de Akaike ("Akaike's Information Theoretic Criterion") este cunoscut sub numele de criteriul AIC şi se poate scrie sub forma (4.19), aproximată de relaţia (4.24) [C25], respectiv (4.25) [J18].

$$AIC(\mathbf{w}) = \log \frac{C_{inv}(\mathbf{w})}{P} + \frac{\Pi \log(P)}{P} \tag{4.24}$$

$$AIC(\Pi) = \log C_{inv} + \frac{2(\Pi + 1)}{P} \tag{4.25}$$

Scanned with OKEN Scanner

❒ ***Criteriu statistic* (III)**

Moody a propus utilizarea numărului efectiv de parametri Π_{ef}, în locul numărului de parametri Π, pentru cazul structurilor regularizate [L38]:

$$\hat{\Gamma} = E_{\text{seturi P}} \{ C_{\text{înv}}(\mathbf{w}) \} \frac{P + \Pi_{ef}}{P - \Pi_{ef}} \tag{4.26}$$

❒ ***Criteriul FPER ("Final Prediction Error criterion for Regularized networks")* (IV)**

Jan Larsen şi Lars Kai Hansen au introdus un alt estimator al erorii medii de generalizare, pentru modele neurale regularizate, numit FPER ("Final Prediction Error criterion for Regularized networks") [L33]:

$$\hat{\Gamma}_{\text{FPER}} = C_{\text{înv}}(\hat{\mathbf{w}}) \frac{P + \hat{\Pi}_1}{P - 2\hat{\Pi}_1 + \hat{\Pi}_2} \tag{4.27}$$

unde parametrii efectivi se determină cu relaţiile (4.28), (4.29). De asemenea, au arătat că, nu este posibilă determinarea unui singur set de parametri efectivi.

$$\hat{\Pi}_1 = \text{tr}\left[\mathbf{H}(\hat{\mathbf{w}})\,\mathbf{A}^{-1}(\hat{\mathbf{w}})\right] \tag{4.28}$$

$$\hat{\Pi}_2 = \text{tr}\left[\mathbf{H}(\hat{\mathbf{w}})\,\mathbf{A}^{-1}(\hat{\mathbf{w}})\,\mathbf{H}\,(\hat{\mathbf{w}})\,\mathbf{A}^{-1}(\hat{\mathbf{w}})\right] \tag{4.29}$$

S-a notat cu $\boldsymbol{A}(\hat{\mathbf{w}}) = \boldsymbol{H}(\hat{\mathbf{w}}) + \boldsymbol{R}$ matricea hessiană a funcţiei cost regularizată (matrice presupusă inversabilă) şi cu $\boldsymbol{H}(\hat{\mathbf{w}})$ matricea hessiană a funcţiei cost $C_{\text{înv}}$:

$$\mathbf{H}(\hat{\mathbf{w}}) = \frac{1}{2} \frac{\partial^2 C_{\text{înv}}(\mathbf{w})}{\partial \mathbf{w}\, \partial \mathbf{w}^T} \tag{4.30}$$

Criteriul FPER poate fi considerat drept o extindere a criteriilor FPE şi GPE ("General Prediction Error" Criterion):

$$\hat{\Gamma}_{\text{GPE}} = C_{\text{înv}}(\hat{\mathbf{w}}) \frac{P + \hat{\Pi}_1}{P - \hat{\Pi}_1} \tag{4.31}$$

Scanned with OKEN Scanner

Din comparaţia criteriului FPER cu FPE şi GPE rezultă că, estimatul cel mai puţin deplasat dintre cele trei este dat de FPER. Din punct de vedere al erorii pătratice medii, estimatorii FPER şi GPE sunt aproximativ identici. Rezultatele cele mai slabe sunt obţinute pentru estimatorul FPE [L33]. Din punct de vedere al dimensiunilor structurilor rezultate, dacă se cunoaşte eroarea de învăţare şi se impune valoarea erorii de generalizare, atunci pentru FPE şi GPE rezultă estimatul numărului parametrilor modelului:

$$\hat{\Pi}_{FPE} = \hat{\Pi}_{GPE} = P\frac{\hat{\Gamma} - C_{\hat{inv}}(\hat{w})}{\hat{\Gamma} + C_{\hat{inv}}(\hat{w})} \tag{4.32}$$

Pentru eroarea de învăţare şi eroarea de test impuse, se poate calcula valoarea numărului parametrilor cu relaţia (4.32) şi se poate compara cu cea obţinută pentru criteriul FPER (relaţiile (4.28) şi (4.29)).

❒ *Criteriul NIC ("Network Information Criterion") (V)*

Problema selecţiei modelului, respectiv determinarea numărului de noduri ascunse prin metode statistice, se poate rezolva prin generalizarea criteriului Akaike pentru modele nerealizabile (semnificaţia termenului este prezentată în continuare). Criteriul rezultat este NIC ("Network Information Criterion") şi a fost propus de Murata, Yoshizawa şi Shun – ichi Amari [M38].

Reţeaua neurală "feedforward" este considerată un model stohastic, care furnizează la ieşire vectorul y atunci când la intrare este prezent vectorul x, pe baza probabilităţii condiţionate $p(\mathbf{w}; y \mid x)$, unde $\mathbf{w}$ este vectorul de parametri ce caracterizează reţeaua.

Se urmăreşte determinarea modelului optim şi a valorii optime a vectorului parametrilor $\mathbf{w}$, pentru aproximarea distribuţiei condiţionate reale din care sunt selectate datele de învăţare. Pentru a rezolva problema statistică dată, se generalizează criteriul AIC în două direcţii:

- ■ Se consideră că distribuţia reală $q(y \mid x)$ nu este neapărat inclusă în nici unul dintre modelele $\{p(\mathbf{w}; y \mid x)\}$. In acest caz, distribuţia reală se numeşte nerealizabilă.
- ■ Se utilizează o funcţie cost generalizată, care include şi un termen de regularizare. Intr-un caz particular, aceasta conduce la estimatorul ML.

Se consideră un model stohastic ce primeşte un vector de intrare $x \in \Re^M$ şi furnizează la ieşire un vector $y \in \Re^N$, cu M şi N dimensiunile spaţiului de intrare, respectiv de ieşire. Vectorul de intrare este generat cu probabilitatea $q(x)$, iar vectorul de ieşire cu probabilitatea condiţionată $q(y \mid x)$. Sistemul este precizat în continuare prin cea din urmă sau, prin probabilitatea reunită intrare – ieşire $q(x, y)$, unde:

Scanned with OKEN Scanner

$$q(\mathbf{x}, \mathbf{y}) = q(\mathbf{x})\, q(\mathbf{y} \mid \mathbf{x}) \tag{4.33}$$

Reţeaua multistrat stohastică este considerată ca o familie parametrică de probabilităţi condiţionate. Probabilitatea condiţionată $p(\mathbf{w}; y \mid x)$, unde $\mathbf{w} \in \mathcal{R}^{\Pi}$, este parametrul Π – dimensional care specifică modelul de reţea. Distribuţia se numeşte realizabilă, iar modelul se numeşte "de încredere" când probabilitatea reală $q(y \mid x)$ aparţine modelului, adică, există $\hat{\mathbf{w}}$ pentru care:

$$q(\mathbf{y} \mid \mathbf{x}) = p(\hat{\mathbf{w}}\,; \mathbf{y} \mid \mathbf{x}) \tag{4.34}$$

Pentru construirea criteriului NIC nu este necesară ipoteza distribuţiei realizabile.

Se poate scrie probabilitatea $p(\mathbf{w}; y \mid x)$ pentru o reţea perceptron multistrat, considerând că funcţia realizată de reţea este $F(\mathbf{w}; x)$. Componentele vectorului de parametri $\mathbf{w}$ sunt ponderi şi praguri. Valorii funcţiei i se adaugă un termen de zgomot $n(x)$ pentru a se obţine valoarea ieşirii reţelei:

$$\mathbf{y} = \mathrm{F}(\mathbf{w}; \mathbf{x}) + \mathbf{n}(\mathbf{x}) \tag{4.35}$$

Zgomotul se numeşte aditiv atunci când distribuţia sa este independentă de x. In acest caz, probabilitatea condiţionată este:

$$p(\mathbf{w}; \mathbf{y} \mid \mathbf{x}) = a(\mathbf{y} - \mathrm{F}(\mathbf{w}; \mathbf{x})) \tag{4.36}$$

unde a este funcţia densitate de probabilitate a zgomotului n. In general, distribuţia $a(n \mid x)$ depinde de x, astfel încât:

$$p(\mathbf{w}; \mathbf{y} \mid \mathbf{x}) = a(\mathbf{y} - \mathrm{F}(\mathbf{w}; \mathbf{x}) \mid \dot{\mathbf{x}}) \tag{4.37}$$

Dacă nu există zgomot, atunci modelul este determinist şi funcţia $a(n \mid x)$ este chiar funcţia delta.

Pentru evaluarea performanţelor reţelei, se defineşte funcţia discrepanţă $D(q, p(\mathbf{w}))$, care măsoară diferenţa dintre distribuţia condiţionată reală $q(y \mid x)$ şi distribuţia condiţionată $p(\mathbf{w}; y \mid x)$ a modelului.

Definiţia 4.1 Funcţia discrepanţă sau funcţia pierdere medie $D(q, p(\mathbf{w}))$ dintre două distribuţii q şi $p(\mathbf{w})$ este media funcţiei de pierdere plus un termen de regularizare [M38]:

$$\mathrm{D}(\mathrm{q}, \mathrm{p}(\mathbf{w})) = \iint \left(\varphi(\mathbf{w}, \mathbf{x}, \mathbf{y}) + \mathrm{r}(\mathbf{w})\right) \mathrm{q}(\mathbf{x}, \mathbf{y})\, \mathrm{dx}\, \mathrm{dy} \tag{4.38}$$

Scanned with OKEN Scanner

unde $r(\mathbf{w})$ este termenul de regularizare. Funcţia pierdere ("loss function") este $\varphi(\mathbf{w}, x, y)$. Dacă aceasta este eroarea medie pătratică, pentru reţeaua perceptron multistrat, se poate scrie relaţia (4.39), care în cazul determinist devine (4.40). O altă alegere posibilă este funcţia dată de relaţia (4.41) sau (4.42).

$$\varphi(\mathbf{w}, \mathbf{x}, \mathbf{y}) = \frac{1}{2} \int \| \mathbf{y} - \mathbf{F}(\mathbf{w}; \mathbf{x}) - \mathbf{n} \|^2 \; \mathbf{a}(\mathbf{n} \mid \mathbf{x}) \; \mathbf{dn} \tag{4.39}$$

$$\varphi(\mathbf{w}, \mathbf{x}, \mathbf{y}) = \frac{1}{2} \| \mathbf{y} - \mathbf{F}(\mathbf{w}; \mathbf{x}) \|^2 \tag{4.40}$$

$$\varphi(\mathbf{w}, \mathbf{x}, \mathbf{y}) = \log \frac{\mathbf{q}(\mathbf{y} \mid \mathbf{x})}{\mathbf{p}(\mathbf{w}; \mathbf{y} \mid \mathbf{x})} \tag{4.41}$$

$$\varphi(\mathbf{w}, \mathbf{x}, \mathbf{y}) = - \log \mathbf{p}(\mathbf{w}; \mathbf{y} \mid \mathbf{x}) \tag{4.42}$$

Pentru cazul funcţiei pierdere eroare pătratică medie, zgomot aditiv şi fără regularizare, expresia (4.38) devine:

$$D(q, p(\mathbf{w})) = \iiint \| \mathbf{y} - \mathbf{F}(\mathbf{w}; \mathbf{x}) - \mathbf{n} \|^2 \; \mathbf{a}(\mathbf{n}) \, q(\mathbf{x}, \mathbf{y}) \; \mathbf{dn} \; \mathbf{dx} \; \mathbf{dy} \tag{4.43}$$

Parametrul optimal, care defineşte modelul optimal în raport cu funcţia discrepanţă sau cu funcţia pierdere medie, depinde de distribuţia reală necunoscută $q(x, y)$. In locul acesteia, se poate utiliza un set de învăţare compus din P exemple, generate de distribuţia reală. Cu alte cuvinte, se poate utiliza distribuţia empirică, construită pe baza celor P exemple de învăţare:

$$q^*(\mathbf{x}, \mathbf{y}) = \frac{1}{\mathrm{P}} \sum_{\xi=1}^{\mathrm{P}} \delta(\mathbf{x} - \mathbf{x}_\xi, \mathbf{y} - \mathbf{y}_\xi) \tag{4.44}$$

unde s-au notat cu (x_ξ, y_ξ) perechile observate intrare – ieşire $(x(\xi), y(\xi))$. Se ştie că, pentru P suficient de mare, $q^*(x, y)$ aproximează $q(x, y)$ şi este rezonabilă evaluarea modelului reţea prin utilizarea $q^*(x, y)$ în loc de $q(x, y)$. Cu toate acestea, pentru un număr finit P de exemple, este necesară evaluarea diferenţei dintre $q(x, y)$ şi $q^*(x, y)$: se notează cu $\mathbf{w}_0$ parametrul reţelei optimale, şi se minimizează funcţia discrepanţă $D(q, p(\mathbf{w}))$:

$$D(q, p(\mathbf{w})) = \min_{\mathbf{w}} D(q, p(\mathbf{w})) \tag{4.45}$$

Scanned with OKEN Scanner

Se procedează similar atunci când distribuţia este cea empirică, notând cu $\mathbf{w}^*$ parametrul cvasi – optimal:

$$D(q^*, p(w^*)) = \min_{w} D\left(q^*, p(w)\right) \tag{4.46}$$

De obicei, w_0 şi $\mathbf{w}^*$ sunt diferiţi şi trebuie evaluată diferenţa acestora.

Pentru evaluarea parametrului cvasi – optimal se aplică metoda stohastică a gradientului negativ.

Definiţia 4.2 La fiecare pas de învăţare, se alege din setul de învăţare un nou exemplu (distribuţia empirică). Parametrul $\mathbf{w}(k)$ la momentul de timp k este modificat astfel [M38]:

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \eta(k) \ \nabla \{\varphi[\mathbf{w}(k), \mathbf{x}(k), y(k)] + r[\mathbf{w}(k)]\} \tag{4.47}$$

unde ∇ marchează gradientul în raport cu parametrul $\mathbf{w}(k)$, iar $\eta(k)$ este rata de învăţare.

In cazul RPM deterministă, relaţia este cea BKP standard:

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \eta \ \{y - F[\mathbf{w}(k), \mathbf{x}]\}^T \ \nabla F[\mathbf{w}(k), \mathbf{x}] \tag{4.48}$$

iar $\mathbf{w}(k) \to \mathbf{w}^*$ pentru $k \to \infty$ şi $\eta \to 0$. Se poate arăta că, parametrul estimat $\mathbf{w}(k)$ are o distribuţie normală, a cărei medie este valoarea parametrului cvasi – optimal $\mathbf{w}^*$, iar dispersia este proporţională cu η.

Pentru formulareą criteriul NIC, se consideră două modele parametrice, având $p_1(\mathbf{w}_1; y \mid x,)$, $\mathbf{w}_1 \in \mathcal{R}^{M_1}$ şi respectiv, $p_2(\mathbf{w}_2; y \mid x)$, $\mathbf{w}_2 \in \mathcal{R}^{M_2}$ $(M_1 < M_2)$. Se presupune că primul model este submodel al celui de-al doilea:

$$\{p_1(\mathbf{w}_1; y \mid x)\} \subset \{p_2(\mathbf{w}_2; y \mid x)\} \tag{4.49}$$

Aceasta implică faptul că, impunând anumite valori unor componente ale parametrului $\mathbf{w}_2$, se obţine primul submodel. In cazul RPM, numărul nodurilor din straturile de intrare şi de ieşire sunt aceleaşi în cele două modele, dar numărul nodurilor ascunse este mai mare în cazul celui de-al doilea model. Impunând condiţia ca ponderile conexiunilor care există în modelul al doilea şi nu există în primul, să aibă valoarea zero, se obţine primul model.

Când se estimează parametrii celor două modele, pe un set comun de date de învăţare, trebuie identificat modelul care este mai bun. Criteriul de alegere

Scanned with OKEN Scanner

poate fi valoarea cea mai mică a funcţiei discrepanţă. Cu toate acestea, distribuţia reală $q(x, y)$ nu este cunoscută, ci numai $q^*(x, y)$, deci se poate estima funcţia discrepanţă şi trebuie evaluată, de asemenea, diferenţa dintre acest estimat şi valoarea reală.

Teorema 4.1 Discrepanţa medie dintre sistemul real $q(x, y) = q(y \mid x)\, q(x)$ şi modelul învăţat $p(\hat{w}; y \mid x)$ pe un set de P exemple este:

$$E\left\{D\left(q, p(\hat{w})\right)\right\} = E\left\{D\left(q^*, p(\hat{w})\right)\right\} + \frac{1}{P}\ \mathrm{tr}\left(\mathbf{G}\,\mathbf{Q}^{-1}\right) + O\left(P^{-\frac{3}{2}}\right) \qquad (4.50)$$

unde $E\{\bullet\}$ marchează operaţia de mediere. S-au notat cu:

$$\mathbf{G} = V_q\ \{\nabla\ [\varphi(\mathbf{w}, \mathbf{x}, \mathbf{y}) + r(\mathbf{w})]\} \qquad (4.51)$$

$$\mathbf{Q} = E_q\ \{\nabla\nabla\ [\varphi(\mathbf{w}, \mathbf{x}, \mathbf{y}) + r(\mathbf{w})]\} \qquad (4.52)$$

unde E_q şi V_q sunt media şi, respectiv, dispersia în raport cu distribuţia având densitatea de probabilitate $q(x)$.

Criteriul NIC Fie $\mathbf{M}_\xi = \{p_\xi(\mathbf{w}_\xi; \mathbf{y} \mid \mathbf{x})\}$ un set de modele ierarhice, date de relaţia:

$$\mathbf{M}_1 \subset \mathbf{M}_2 \subset \mathbf{M}_3 \subset \ldots \qquad (4.53)$$

unde $\mathbf{M}_\xi$ este un submodel al $\mathbf{M}_\psi$ ($\xi < \psi$). Fie $\hat{\mathbf{w}}_\xi$ parametrul modelului $\mathbf{M}_\xi$, obţinut prin învăţarea pe un set comun de P patternuri. Criteriul NIC este:

$$\mathrm{NIC}(p_\xi) = D\left(q^*, p_\xi\left(\hat{\mathbf{w}}_\xi\right)\right) + \frac{1}{P}\ \mathrm{tr}\left(\mathbf{G}^*_\xi\left(\hat{\mathbf{w}}_\xi\right)\mathbf{Q}^*_\xi\left(\hat{\mathbf{w}}_\xi\right)^{-1}\right) \qquad (4.54)$$

Modelul care minimizează criteriul NIC este optimal din punct de vedere al valorii medii a funcţiei pierdere, adică a discrepanţei medii $E\left\{D\left(q, p_\xi\left(\hat{\mathbf{w}}_\xi\right)\right)\right\}$. S-au notat cu:

$$\mathbf{G}^*_\xi\left(\hat{\mathbf{w}}_\xi\right) = V_{p^*}\left[\nabla\, \varphi\left(\hat{\mathbf{w}}_\xi, \mathbf{x}, \mathbf{y}\right)\right] \qquad (4.55)$$

$$\mathbf{Q}^*_\xi\left(\hat{\mathbf{w}}_\xi\right) = E_{p^*}\left[\nabla\nabla\, \varphi\left(\hat{\mathbf{w}}_\xi, \mathbf{x}, \mathbf{y}\right)\right] \qquad (4.56)$$

Scanned with OKEN Scanner

Comparaţia realizată între rezultatele criteriului NIC şi rezultatele obţinute prin aplicarea altor criterii conduce la următoarele concluzii:

■ Dacă modelul este "de încredere" şi funcţia pierdere este logaritmul negativ, atunci problema se reduce la selectarea modelului statistic pentru estimarea distribuţiei reunite $q(x, y)$. In acest caz $\boldsymbol{G} = \boldsymbol{Q}$ este matricea de informaţie Fisher şi,

$$\mathrm{tr}\left(\mathbf{G}_{\xi}^{*}\ \mathbf{Q}_{\xi}^{*\ -1}\right) = \Pi_{\xi} + O\left(\frac{1}{\sqrt{P}}\right) \tag{4.57}$$

unde Π_{ξ} este numărul parametrilor modelului $\mathbf{M}_{\xi}$. In acest caz, criteriul NIC se obţine prin împărţirea relaţiei corespunzătoarea criteriului AIC la $2P$. Deci, NIC este o generalizare a criteriului AIC.

■ Criteriul NIC, asemeni criteriului AIC, se poate utiliza pentru selectia modelelor dintr-o ierarhie de modele, în care un submodel este inclus în altul.

❒ *Criteriul PMDL ("Predictive Minimum Description Length")* (VI)

Pe baza studiilor lui Akaike legate de alegerea modelului, Rissanen a propus utilizarea codului de lungime minimă pentru datele disponibile (se consideră minimul necesar de informaţie, pentru descrierea datelor [G17]) drept criteriu pentru alegerea modelului [L35, L36].

Complexitatea statistică a modelului poate fi considerată generalizarea informaţiei lui Shannon, I_S [L35]. Pentru o distribuţie dată $p(\boldsymbol{x})$, logaritmul său cu semn schimbat, evaluat pentru datele disponibile (observate) $\boldsymbol{x}$, este (4.58) şi se mai numeşte complexitatea lui Shannon a datelor relativ la o distribuţie dată, sau clasă de modele $\mathbf{M} = \{p(\boldsymbol{x})\}$. Clasa de modele luată în considerare este (4.59), unde Ω^{Π} este un subset nevid al spaţiului Π – dimensional euclidian, iar Π este numărul parametrilor modelului (reţelei). Notaţiile p şi a se referă la funcţii densitate de probabilitate.

$$\mathrm{I_S} = -\log(\mathrm{p}(\mathbf{x})) \tag{4.58}$$

$$\mathbf{M} = \{\mathrm{p}(\mathbf{x} \mid \mathbf{w}), \mathrm{a}(\mathbf{w}) \mid \mathbf{w} \in \Omega^{\Pi}, \Pi = 1, 2, \ldots\} \tag{4.59}$$

Complexitatea stohastică a datelor $\boldsymbol{x}$, relativ la clasa de modele $\mathbf{M}$, este definită [L35, L36] ca:

$$\mathrm{I}(\mathbf{x} \mid \mathbf{M}) = -\log \mathrm{p}(\mathbf{x} \mid \mathbf{M}) = -\log \int_{\mathbf{w} \in \Omega^{\Pi}} \mathrm{p}(\mathbf{x} \mid \mathbf{w})\ \mathrm{da}(\mathbf{w}) \tag{4.60}$$

Scanned with OKEN Scanner

Deși clasa de modele **M** include distribuția "apriori" a, aceasta nu are același rol ca în teoria bayesiană. Nu trebuie aleasă, de vreme ce se poate construi o distribuție $a(\mathbf{w} \mid x)$ proporțională cu $p(x \mid \mathbf{w})$. Complexitatea stohastică reprezintă atunci cel mai scurt cod pe care îl poate obține o clasă de modele dată.

In cazul rețelelor neurale cu un singur nod de ieșire ($N = 1$), utilizate pentru probleme de predicție, modelul nu este ales ca distribuție, ci de forma:

$$\hat{x}(k+1) = F(\mathbf{x}(k) \mid \mathbf{w}) \tag{4.61}$$

unde $\hat{x}$ este valoarea obținută prin predicție la momentul $k + 1$; $x(k)$ reprezintă vectorul de date de intrare la momentul k, având componentele $[x_{k-M}, \ldots x_k]$; $F_{\mathbf{w}}$ este funcția neliniară globală realizată de rețea, iar **w** este vectorul parametrilor (ponderi, praguri).

Eroarea obținută la ieșire la momentul k, este dată de relația (4.62). Pentru problema de predicție considerată, $\hat{x}(k)$ este valoarea obținută prin predicție la momentul k, pe baza valorilor anterioare și este dată de (4.63).

$$e_k \overset{\text{def.}}{=} e(k) = x(k) - \hat{x}(k) \tag{4.62}$$

$$\hat{x}_k \overset{\text{def.}}{=} \hat{x}(k) = F_{\mathbf{w}}[\mathbf{w}; x_1, x_2, \ldots x_{k-1}] \tag{4.63}$$

Modelul (4.63) poate fi formulat ca model stohastic, definind distribuția gaussiană condiționată:

$$p(\mathbf{w}; x_{k+1} \mid \mathbf{x}_k ; \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{e_{k+1}^2}{\sigma^2}\right) \tag{4.64}$$

cu $x_k = [x_1, x_2, \ldots x_k]$ și σ^2 dispersia datelor de la intrarea rețelei. Densitatea de probabilitate din relația (4.64) este extinsă în cazul secvențelor, prin multiplicarea cu expresia:

$$-\ln p\left(\mathbf{w}; x_{k+1} \mid \mathbf{x}_k ; \sigma^2\right) = \frac{1}{2\sigma^2} \sum_{k=0}^{P-1} e_{k+1}^2 + \frac{P}{2} \ln\left(2\pi\sigma^2\right) \tag{4.65}$$

După alegerea clasei modelelor, se estimează lungimea celui mai scurt cod care se poate obține pentru clasa respectivă. Se folosesc notațiile (4.66) pentru estimatele ML ale parametrilor care minimizează lungimea codului (4.67), pentru datele anterioare.

Scanned with OKEN Scanner

$$\hat{\mathbf{w}}_k \overset{\text{not.}}{=} \hat{\mathbf{w}}(\mathbf{x}_k), \quad \hat{\sigma}_k \overset{\text{not.}}{=} \hat{\sigma}(\mathbf{x}_k) \tag{4.66}$$

$$-\ln p\left(\mathbf{w}; x_{k+1} \mid \mathbf{x}_k; \sigma^2\right) \tag{4.67}$$

In particular, pentru cazul relației (4.68) rezultă lungimea codului dată de (4.69). Valoarea inițială $\hat{\sigma}_0^2$ se alege.

$$\hat{\sigma}_k^2 = \frac{1}{k} \sum_{i=1}^{k} e_i^2 \tag{4.68}$$

$$-\ln p(\mathbf{x}_P \mid \Pi) = \frac{1}{2} \sum_{k=0}^{P-1} \left[\frac{e_{k+1}^2}{\hat{\sigma}_k^2} + 2 \ln \hat{\sigma}_k \right] + \frac{P}{2} \ln(2\pi) \tag{4.69}$$

Criteriul poate scris și sub forma (4.21), cu P numărul total de observații.

Costul modelului, deci lungimea codului necesar modelului, apare numai implicit. Parametrii rețelei nu trebuie codați, ci se determină din șirul de date anterior, pe baza algoritmului. Costul modelului PMDL se adaugă la erorile de predicție, iar caracteristicile de supraînvățare și de subînvățare sunt penalizate automat.

Algoritmul PMDL este prezentat în tab. 4.4 [L36]. S-a considerat eroarea de predicție dată de relația:

$$C_{(k,\,d)} = C_{(k-1,\,d)} + \sum \left[x(k+1) - \hat{x}(k+1) \right]^2 \tag{4.70}$$

Valorile numărului de noduri de intrare i și cel al numărului de noduri ascunse h sunt modificate, calculându-se eroarea de predicție pentru fiecare combinație (i, h). In final, se selectează combinația pentru care eroarea este minimă. Pentru problema predicției seriilor de timp, cu M și H în intervalul $M_{min} \le M \le M_{max}$, $H_{min} \le H \le H_{max}$, se poate calcula valoarea erorii de predicție și se determină perechile (M, H) pentru care aceasta este minimă.

4.4.4 Comparație a criteriilor statistice considerate pentru proiectarea structurilor neurale "feedforward"

Comparații cantitative ale rezultatelor aplicării diferitelor criterii statistice pentru proiectarea structurilor neurale "feedforward" sunt prezentate de Hagiwara și Usui [H39], bazate pe experimente numerice.

Scanned with OKEN Scanner

Tab. 4.4 Algoritmul PMDL

❶ Se generează şirul de date x_p, de lungime P.

❷ Se împarte şirul de date x_p într-un număr de:

$n_{max} = 1 + [(P - d_1) / d]$

segmente consecutive de lungime d, excepţie făcând numai segmentul iniţial, de lungime d_1.

❸ Se stabileşte eroarea iniţială de predicţie, C_0 (de exemplu, $C_0 = 0$).

❹ Pentru $M_{min} \le i \le M_{max}$ (noduri de intrare)

Pentru $H_{min} \le h \le H_{max}$ (noduri ascunse)

Se stabilesc valorile iniţiale ale ponderilor;

Pentru $1 \le k \le n_{max} - 1$ (numărul segmentelor de date)

- Reţeaua învaţă segmentul de date k;
- Se calculează eroarea de predicţie C_k, pentru segmentul de date $k+1$ utilizat pentru învăţare;
- Eroarea de predicţie totală C_{total} este:

$C_{total} \leftarrow C_{total} + C_k$

☐ $k = k + 1$;

- $C_{(i,h)} \leftarrow \dfrac{C_{total}}{P - d_1}$

☐ $h = h + 1$;

☐ $i = i + 1$;

❺ Se determină valoarea $C_{(i,h)}$ minimă, dintre valorile obţinute.

☐ *Comparaţie a criteriilor statistice* (I)

Se consideră structura RPM cu M intrări şi H noduri ascunse, în cazurile $M = 1$ şi $M > 1$. Concluziile aplicării criteriilor statistice pentru estimarea dispersiei unei distribuţii normale sunt:

■ Estimatul AIC şi TIC pentru complexitatea reţelei (evaluată de $C(\Pi)$) are valoare mică, iar pentru BIC / MDL şi PMDL are valoare mare (s-a notat cu BIC criteriul clasic Bayes). Concluzia se bazează pe trasarea histogramei pentru frecvenţa de selectare a nodurilor ascunse, în funcţie de numărul acestora. Rezultă că, BIC / MDL (dacă setul de date este mare, atunci şi PMDL), tind să selecteze structuri simple, în timp ce AIC selectează reţeaua cu dimensiuni mari sau chiar maxime (fig 4.2 şi 4.3).

Scanned with OKEN Scanner

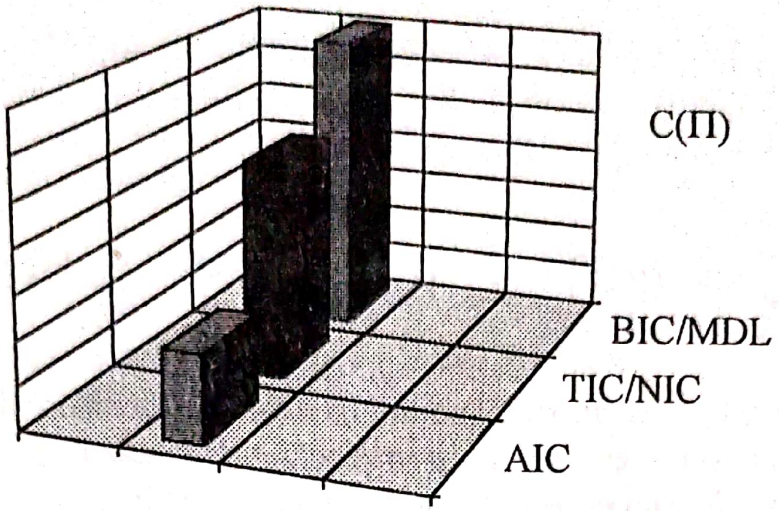


Fig. 4.2 Complexitatea $C(\Pi)$ (relaţia 4.16) în funcţie de numărul nodurilor ascunse (pentru H dat)

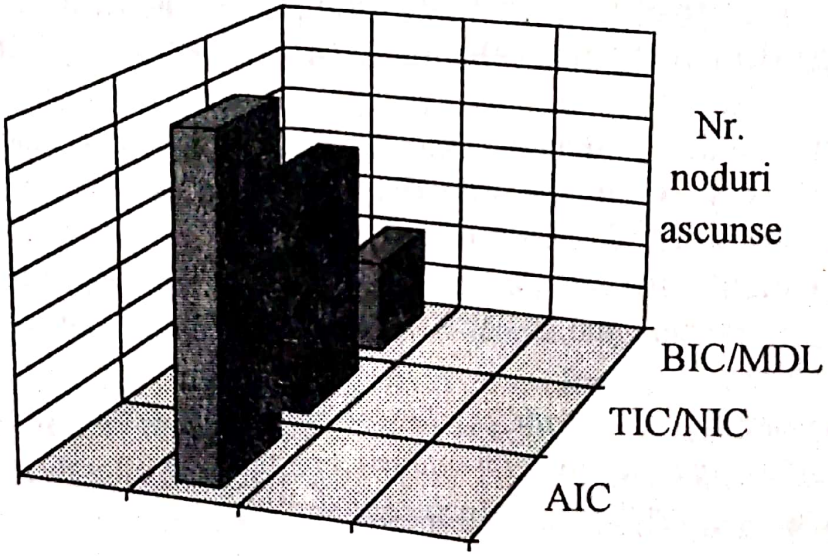


Fig. 4.3 Dimensiunea reţelei selectate (numărul nodurilor ascunse) prin aplicarea criteriilor statistice

❒ ***Comparaţie a criteriilor statistice* (II)**

Gish abordează probleme de clasificare, pentru cazul numărului claselor egal cu 2 [G7]. Reţeaua realizează funcţia $F(\mathbf{w}; \boldsymbol{x})$, unde $\mathbf{w}$ sunt parametrii modelului. Se urmăreşte selectarea acestora, astfel încât $F(\mathbf{w}; \boldsymbol{x})$ să fie un estimat

al probabilităţii $p_w(C_1 \mid x)$, ca vectorul de intrare x este clasificat în clasa C_1, fiind observat x.

Funcţia cost obţinută prin aplicarea criteriului verosimilităţii maxime este (4.71) şi poate fi utilizată pentru exprimarea criteriilor AIC şi MDL sub forma (4.19), (4.20), cu P numărul total de observaţii.

$$\log C = \sum_{x \in C_1} \log F(w; x) + \sum_{x \in C_2} \left[1 - \log F(w; x)\right] + \sum_{\xi=1}^{P} \log p(x_\xi) \qquad (4.71)$$

Gish a arătat că, pentru probleme de clasficare, criteriul MDL conduce la modele cu un număr mai mic de parametri decât AIC în cazul în care numărul de patternuri este $P > P_0$ (s-a considerat ca exemplu $P_0 = 8$).

❒ ***Comparaţie a criteriilor statistice* (III)**

Comparaţia estimatului FPER cu GPE şi AIC [L33] arată că, FPER conduce la estimatul cel mai puţin deplasat dintre cele trei.

Din punct de vedere al erorii pătratice medii, estimatorii FPER şi GPE sunt aproximativ identici. Rezultatele cele mai slabe sunt obţinute pentru estimatorul FPE.

Din punct de vedere al dimensiunilor structurilor rezultate, dacă se cunoaşte eroarea de învăţare şi se impune valoarea erorii de generalizare, atunci pentru FPE şi GPE rezultă numărul estimat al parametrilor.

Din comparaţiile realizate rezultă că, dintre criteriile prezentate, AIC conduce la rezultatele cele mai slabe. Rezultate mai bune se obţine prin aplicarea GPE, NIC şi FPER.

Este important faptul că, oricare dintre criteriile statistice prezentate, poate constitui un criteriu de oprire, atât pentru creşterea (cap. 4.5), cât şi pentru descreşterea (cap. 4.6) reţelelor ontogenice "feedforward".

4.5 METODE ONTOGENICE CONSTRUCTIVE

Fiind disponibil un set de date de învăţare, proiectarea unei RPM se poate realiza prin creşterea numărului neuronilor / straturilor ascunse, până la îndeplinirea unui criteriu impus. Abordarea se numeşte "constructivă", deoarece structura finală este mai mare decât cea iniţială.

Au fost propuşi o serie de algoritmi constructivi, cu performanţe diferite din punct de vedere al complexităţii structurii rezultate, capacităţii de generalizare, costului proiectării structurii, timpului necesar proiectării, etc.

Scanned with OKEN Scanner

4.5.1 Formularea problemei

Problema creşterii structurii neurale poate fi formulată ca o căutare în spaţiul funcţional. Se doreşte determinarea dimensiunilor reţelei care aproximează funcţia dorită. Spaţiul funcţional, stările iniţială şi finală, precum şi strategia de căutare sunt specifice fiecărei metode constructive.

4.5.1.1 Spaţiul de căutare

Spaţiul de căutare (spaţiul funcţional) este un subspaţiu într-un spaţiu liniar normat[13] dat. Se poate defini noţiunea de distanţă: dacă **w** şi **v** se află în spaţiul liniar normat, atunci distanţa de la **w** la **v** (sau de la **v** la **w**) este $\| \mathbf{v} - \mathbf{w} \|$.

In practică nu se poate realiza o căutare completă în tot spaţiul funcţional. De aceea, spaţiul de căutare, ce conţine toate funcţiile realizate de reţea care pot fi implementate de un algoritm constructiv, este numai un subspaţiu al acestuia şi este determinat de modul în care sunt generate nodurile ascunse. Acesta este definit prin formularea răspunsurilor la următoarele probleme:

- Când trebuie adăugat un nou perceptron simplu ascuns ?
- Cum se conectează noul perceptron simplu ascuns în structură ?
- Care este funcţia de activare a perceptronului simplu ascuns adăugat ?
- Cum se determină ponderile perceptronului simplu ascuns adăugat ?

4.5.1.2 Starea iniţială şi starea finală

Deseori, în starea iniţială, structurile nu conţin nici un nod ascuns şi există sau nu conexiuni directe intrare – ieşire.

Starea finală este evaluată în funcţie de criteriul de cost ales (de exemplu, norma L_p). Există mai multe structuri finale posibile, care realizează aproximarea satisfăcătoare a funcţiei dorite. Metodele constructive disponibile aleg dintre acestea pe cea care realizează o aproximare "suficient de bună" a funcţiei dorite.

4.5.1.3 Strategia de căutare

Strategia de căutare este specifică fiecărei metode ontogenice constructive. Se generează o structură neurală şi se testează conform criteriului ales. Dacă acesta este îndeplinit, atunci procesul s-a încheiat, altfel se reia generarea structurii.

---

[13] Un spaţiu liniar normat este un spaţiu liniar, definit împreună cu o funcţie reală $\| \cdot \|$ numită normă. Pentru orice **v** din spaţiu, norma satisface cele trei proprietăţi [K33]:

① $\| \mathbf{v} \| \geq 0$, egalitatea având loc numai dacă $\mathbf{v} = 0$.

② $\| \lambda \mathbf{v} \| = \|\lambda\| \, \| \mathbf{v} \|$ pentru orice scalar λ.

③ $\| \mathbf{v} + \mathbf{w} \| \leq \| \mathbf{v} \| \, \| \mathbf{w} \|$.

Scanned with OKEN Scanner

Din punct de vedere al structurii rezultate, problemele la care se formulează răspunsuri sunt:

- Care este tipul arhitecturii rezultate (RPM, rețea RBF, etc.) ?
- Structura finală are un singur strat ascuns sau mai multe ?
- Arhitectura rezultată este total sau parțial interconectată ? Există conexiuni numai între straturi succesive, sau și alte conexiuni între straturi ?

Se urmărește obținerea unei structuri cât mai reduse, cu un cost de proiectare scăzut și care să poată generaliza corespunzător. Stabilirea unui compromis între deplasament și dispersie este esențială, pentru că primul scade, iar cea de-a doua crește odată cu adăugarea unor noi noduri ascunse [D205]. De aceea, întrebărilor formulate anterior li se adaugă următoarea:

- Când trebuie oprit procesul de creștere a numărului neuronilor ascunși ?

4.5.2 Clasificarea metodelor ontogenice constructive

O sinteză a metodelor constructive în general este realizată de Hu [H30], iar pentru probleme de regresie[14] în special, de Kwok și Yeung [K33]. Problemele de clasificare reprezintă un caz particular al acestora[15]. Există însă metode, care pot fi aplicate numai problemelor de clasificare și vom preciza acest lucru în fiecare caz.

Există mai multe criterii de clasificare a metodelor ontogenice constructive: strategia constructivă adoptată (tab. 4.5), funcția de activare a nodului ascuns adăugat (tab. 4.6), arhitectura rezultată (tab. 4.7), problema de studiu abordată și criteriul de oprire a creșterii rețelei.

Nodurile ascunse adăugate sunt numite parametrice, dacă funcția de activare este fixă și neparametrice, dacă aceasta nu este restrânsă la o anumită clasă de funcții. Interconectarea este totală dacă toate nodurile dintre două straturi succesive sunt conectate între ele și parțială altfel. Cea din urmă poate fi obținută prin metode care conduc la o arhitectură RPM multistrat obișnuită, sau prin competiție: noul nod ascuns este ales dintr-o clasă de candidați, fiecare fiind conectat diferit în rețeaua neurală.

Din punct de vedere al problemei de studiu abordate, o caracteristică a majorității metodelor constructive este aceea că, spre deosebire de celelalte tipuri de metode, utilizează ca problemă de test ("benchmark"), mai ales problema "două spirale", ceea ce face posibilă comparația diferiților algoritmi pe aceeași aplicație.

---

[14] In probleme de regresie, se dă un vector de intrare oarecare x, M – dimensional și un vector răspuns d. Funcția F care descrie relația generală dintre variabilele x, y se numește suprafață de regresie. In metodele constructive, se urmărește găsirea rețelei care realizează o reprezentare bună a funcției F, altfel spus, este o aproximare convenabilă a funcției F [K33].

[15] De exemplu, pentru o problemă de clasificare cu N clase, se poate construi în principiu o rețea neurală cu N noduri de ieșire, ale căror valori de ieșire sunt de obicei în intervalul [0, 1]. Nodul de ieșire n ($1 \le n \le N$) corespunde clasei C_n și învață probabilitatea aposteriori a clasei C_n, când se cunoaște intrarea x [K33].

Scanned with OKEN Scanner

Tab. 4.5 Clasificarea metodelor crescătoare după strategia constructivă

Metode ontogenice crescătoare
Ad hoc
Metode de partiţionare a spaţiului de intrare
Metode de corectare a erorii
Metode care generează arbori
Metode care construiesc reţeaua pornind de la un arbore
Metode de modularizare a reţelei

Tab. 4.6 Clasificarea metodelor ontogenice crescătoare după funcţia de activare a nodului ascuns adăugat

Funcţia de activare a nodului ascuns adăugat		Tipul reţelei
f(x)	ϕ(net)	RPM (Reţea perceptron multistrat)
$\phi(\mathbf{w}^T \mathbf{x} + b)$	$\frac{1}{1+\exp(-\text{net})}$	
	tanh(net)	
$\phi(\|\mathbf{x} - \mathbf{x}_i\|)$	Funcţie nucleu, deseori gaussiană: $\exp\left(-\text{net}^2/2\sigma^2\right)$	RBF (Reţea cu funcţii de bază radială)
$\sum_{r=1}^{R} c_r h_r\left(\mathbf{w}^T \mathbf{x}\right)$ h_r sunt funcţii Hermite, R este ordinul acestora		PPL ("Projection Pursuit Learning")
Fiecare nod ascuns este o reţea sigma – pi: $\prod_{i=1}^{k} \left(\mathbf{w}_i^T \mathbf{x} + b_i\right)$ unde k este ordinul său		RPN ("Ridge Polynomial Networks")

Tab. 4.7 Clasificarea structurilor neurale rezultate prin metode constructive

Structuri neurale rezultate prin metode constructive		
Un singur strat ascuns	Noduri ascunse simple	
	Noduri ascunse flexibile	Neparametrice
		Parametrice
Mai multe straturi ascunse	Total conectate la toate nodurile existente	
	Parţial conectate la nodurile existente	Conectare fixă
		Conectare prin competiţie

Scanned with OKEN Scanner

Alegerea criteriului de oprire se poate baza pe metodele statistice, care asigură obţinerea generalizării dorite. In acest caz, clasificarea devine cea din cap. 4.4.

In continuare vom utiliza clasificarea propusă în tab. 4.5.

4.5.3 Metode ad hoc

Metodele ad hoc adaugă câte un nou perceptron simplu ascuns în timpul procesului de învăţare, atunci când eroarea nu mai scade.

❒ ***Metodă ad hoc* (I)**

Ash [N12], a propus adăugarea unui nou perceptron simplu ascuns în reţea, atunci când eroarea nu mai scade un interval de timp, în cursul învăţării. Cu toate acestea, stabilirea timpului de aşteptare este subiectivă, la alegerea proiectantului. Odată adăugat un nou nod ascuns, toate ponderile sunt modificate, iar procesul de învăţare trebuie reluat.

❒ ***Metodă ad hoc* (II)**

Chang [H30], a propus o metodă de clasificare a vectorilor binari. Când tendinţa de reducere a erorii de învăţare se micşorează, se adaugă un nou perceptron simplu ascuns, ale cărui ponderi sunt iniţializate astfel încât să fie corectată o eroare de clasificare a reţelei. Apoi, se reia procesul de învăţare. S–a arătat că, această metodă garantează reducerea erorii de învăţare sub un prag prestabilit.

❒ ***Metoda G* (*"Generation"*) (III)**

Hanovar şi Uhr [F28], propun o schemă de creştere a reţelei, în care permite unui întreg con de recunoaştere din componenţa reţelei să–şi modifice topologia, atunci când eroarea nu mai scade în timpul învăţării.

❒ ***Metoda "Tiling"* (IV)**

Mézard şi Nadal [H19], au realizat un algoritm de "acoperire" ("tiling"), care adaugă straturi şi noduri în cadrul stratului, până se obţine convergenţa. Fiecare nou strat conţine mai puţine noduri decât cel precedent (fig. 4.4), astfel încât procesul se încheie atunci când se obţine un singur nod de ieşire.

Scanned with OKEN Scanner

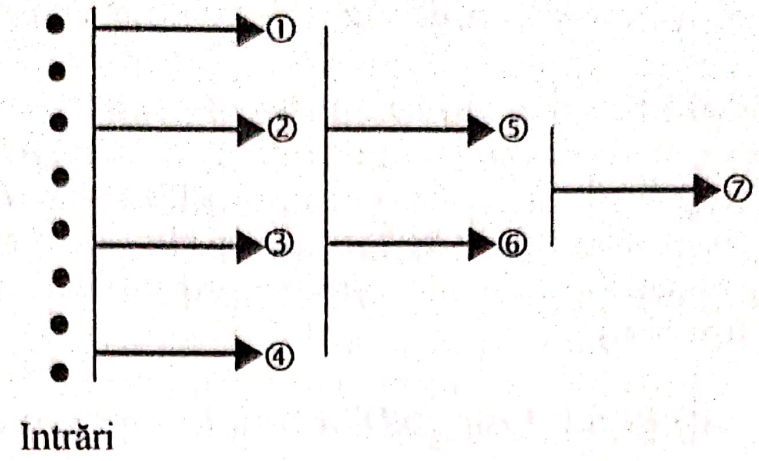


Fig. 4.4 Construirea structurii prin algoritmul "Tiling". Nodurile sunt numerotate în ordinea introducerii lor, iar săgeţile reprezintă conexiunile de la toate nodurile aflate la baza săgeţii

❒ ***Metoda Lee – Tenorio* (V)**

Din relaţia de definiţie (4.72), a erorii empirice generalizate minime EMGM ("Generalized Minimum Empirical Error"), propusă de Lee şi Tenorio, se poate determina numărul nodurilor ascunse H. S-a notat cu dimVC dimensiunea Vapnik – Chervonenkis [H30] a clasificatorului perceptron multistrat, P este numărul eşantioanelor din setul de învăţare, iar λ este un factor de ponderare dependent de aplicaţie.

$$EMGM = C_{\hat{inv}} + \lambda \sqrt{\frac{\dim VC}{P} \ln\left(\frac{P}{\dim VC}\right)} \qquad (4.72)$$

Cel de-al doilea termen este un estimat al erorii de generalizare C_g. Pentru a identifica clasificatorul optimal, Lee şi Tenorio au propus utilizarea valorii $H = 1$ şi apoi creşterea acesteia cu câte o unitate. La fiecare pas de învăţare a reţelei, se calculează şi valoarea EMGM corespunzătoare. Se consideră valoare optimă H cea pentru care se obţine EMGM minimă [H30].

4.5.4 Metode de partiţionare a spaţiului de intrare

Această clasă de metode consideră că, ieşirile perceptronilor simpli din stratul ascuns realizează o reprezentare internă a datelor de intrare, care au fost anterior transformate neliniar. Vectorii de intrare care activează un anumit nod

Scanned with OKEN Scanner

ascuns sunt clasificaţi (grupaţi) împreună, în clasa asociată acelui nod. Numărul nodurilor ascunse reprezintă în acest caz, numărul partiţiilor realizate până în acel moment.

Când la intrare se prezintă un nou eşantion din setul de învăţare, acesta va fi clasificat într–una dintre partiţiile existente. Dacă acest lucru se realizează corect, atunci configuraţia reţelei rămâne aceeaşi, altfel se adaugă un nou neuron ascuns, reprezentând o nouă clasă. Pe baza acestei idei, se pot alege multe dintre tehnicile de partiţionare existente, precum cuantizarea vectorilor, teoria rezonanţei adaptive (ART).

❒ *Metoda RCE ("Restricted Coulomb Energy")* (I)

Reţeaua RCE ("Restricted Coulomb Energy") este o structură RPM cu trei straturi, dintre care două sunt active. Stratul ascuns este constituit din neuroni cu funcţie de activare gaussiană. Neuronii din stratul de ieşire sunt activaţi numai dacă cei din stratul precedent, la care sunt direct conectaţi, sunt activi.

Reţelele de acest tip, în care sunt prezenţi neuroni cu funcţii de activare nemonotone, se mai numesc reţele cu funcţii de bază radială (RBF — "Radial Basis Functions").

Metoda de creştere a reţelei RCE [N12] este o metodă crescătoare în care fiecare nod ascuns este asociat unui număr de prototipuri. Un prototip este activat dacă distanţa dintre acesta şi un nou vector de intrare este mai mică decât o valoare predefinită. Dacă nici unul dintre nodurile ascunse existente nu este activat, atunci se adaugă un nou perceptron simplu ascuns. După acest pas, valoarea prestabilită (prag) a prototipurilor existente se modifică în mod corespunzător.

❒ *Metoda GAL ("Grow and Learn")* (II)

Metoda GAL ("Grow And Learn" = "creşte şi învaţă") [A501] are la bază un algoritm ce foloseşte reprezentarea locală. Perceptronii simpli ascunşi concurează pentru a realiza clasificarea. Este selectat cel mai bun dintre aceştia (activat astfel încât valoarea ieşirii sale să fie 1), iar cei rămaşi sunt inhibaţi (ieşire 0). Dacă în continuare are loc o clasificare eronată, atunci se adaugă un nou nod ascuns şi toate ponderile sunt modificate astfel încât, noul perceptron simplu să fie activat.

In algoritmul competitiv este inclusă şi inhibiţia laterală, pentru a mări performanţele învăţării. De asemenea, este propusă şi o etapă de "somn" ("sleep procedure") pentru a înlătura redundanţa dintre nodurile ascunse. Un nod ascuns este înlăturat dacă reţeaua obţine acelaşi rezultat de clasificare cu sau fără acel nod.

Metoda GAL se aplică numai pentru probleme de clasificare [K33].

Scanned with OKEN Scanner

❒ ***Algoritmul OSA ("OilSpot Algorithm") (III)***

Algoritmul OSA ("Oilspot Algorithm") propus de Frattale Mascioli, Martinelli, ş.a. [M59], operează cu date binare în spaţiul de intrare. Se determină hiperplanele de separare realizate de perceptroni simpli, luând în considerare proprietăţile topologice ale nodurilor hipercubului ce corespunde setului de învăţare. Algoritmul transformă datele reale de intrare în date binare, printr-o codificare ce păstrează vecinătăţile (şi anume, două puncte alăturate în spaţiul de intrare real, trebuie codate în două noduri apropiate ale hipercubului multidimensional). Perceptronul cu două straturi rezultat conţine un strat ascuns şi un strat de ieşire. Cel din urmă are un singur nod şi realizează funcţia logică SAU.

Algoritmul nu necesită setarea anumitor parametri sau stabilirea criteriului de oprire a învăţării. Pe de altă parte, un dezavantaj este faptul că, codarea determină creşterea dimensiunii intrării (de exemplu, de la spaţiul bidimensional la cel octodimensional, pentru codare cu 8 biţi).

4.5.5 Metode de corectare a erorii

Atunci când la intrarea reţelei se prezintă un nou eşantion din setul de date de învăţare şi la ieşire se obţine o valoare mare a erorii, se poate realiza unul dintre paşii următori:

- Se adaugă un nou perceptron simplu ascuns, pentru a putea reprezenta noul eşantion de date de intrare
- Se modifică ponderile, astfel încât noul eşantion să fie reprezentat, chiar cu riscul de a uita eşantioane învăţate anterior.

Din punct de vedere al modului de realizare a conectării noilor noduri / straturi ascunse în reţea, acestea pot fi conectate total sau parţial la nodurile existente anterior.

4.5.5.1 Metode ce interconectează total elementele introduse în reţea, cu cele existente

❒ ***Metoda CASCOR ("Cascade Correlation") (I)***

Metoda CASCOR ("Cascade Correlation") a fost propusă de Scott Fahlman [F18] şi construieşte o structură "feedforward", care evoluează de la o dimensiune iniţială minimă, prin adăugarea câte unui nou strat, care conţine un singur nod ascuns, atunci când nu se poate realiza învăţarea. Ponderile tuturor nodurilor ascunse inserate anterior sunt îngheţate. Învăţarea este supervizată.

Învăţarea în reţeaua CASCOR are loc în doi paşi:

Scanned with OKEN Scanner

❶ Ponderile de la intrarea noului nod sunt calculate, prin maximizarea modulului covariaţiei.

❷ Ponderile conexiunilor ce leagă atât nodurile ascunse de ieşiri, cât şi intrările de ieşiri, sunt optimizate, prin minimizarea erorii pătratice minime.

Structura iniţială conţine numai noduri de intrare şi noduri de ieşire (fig. 4.5). Fiecare intrare este conectată la fiecare ieşire şi există, de asemenea, o intrare de prag, a cărei valoare este +1.

Se modifică ponderile acestor conexiuni intrare – ieşire, prin învăţarea setului de date de învăţare. Se poate utiliza algoritmul Quickprop [F18] care, în absenţa nodurilor ascunse, devine în esenţă legea de învăţare "delta", dar cu viteză de convergenţă mai ridicată.

Când nu se mai înregistrează scăderi semnificative ale erorii, se mai trece o dată întregul set de date de învăţare prin reţea, pentru determinarea valorii erorii. Dacă aceasta este mai mică decât un prag, atunci învăţarea se opreşte. Altfel, se adaugă un nou nod ascuns (fig. 4.6).

Pentru aceasta, se alege un nod "candidat", conectat la toate intrările reţelei şi la toate nodurile ascunse anterioare. Ieşirea sa încă nu este conectată în reţea. Acesta primeşte semnale de intrare de la întregul set de intrări în reţea, precum şi de la nodurile ascunse precedente. Pentru fiecare nod, valoarea ieşirii este dată de relaţia (4.73), unde notaţiile au semnificaţia menţionată în par. 3.2.1.

$$z_h(\xi) = \tanh\left(\sum_{i=1}^{M+H} w_{ih}\, x_i(\xi)\right) \qquad (4.73)$$

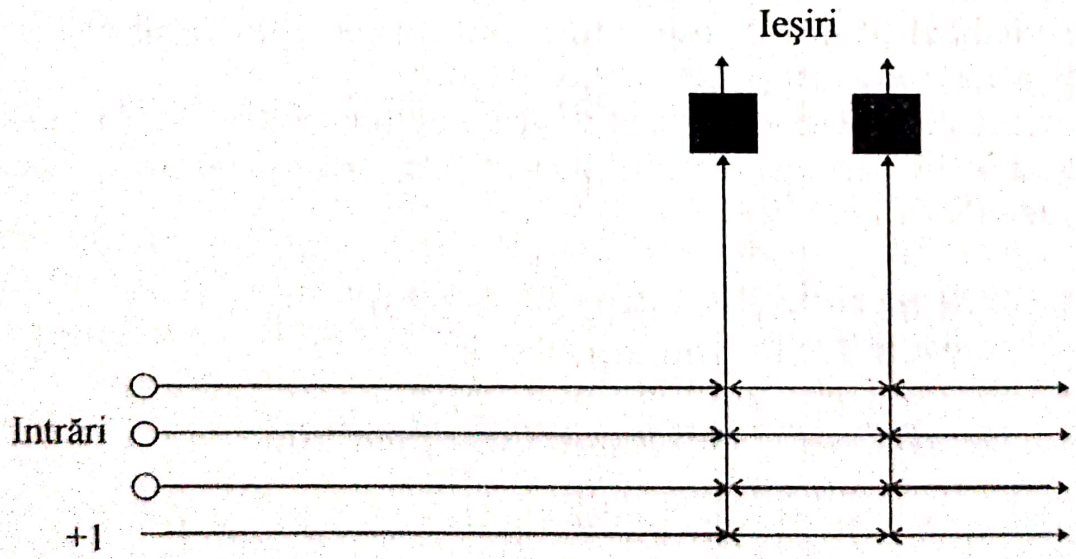


Fig. 4.5 Structura iniţială a reţelei CASCOR. Activările se sumează pe liniile verticale. Conexiunile marcate X se modifică prin învăţare. Nodurile marcate ■ sunt noduri active (funcţie de activare *tanh*)

Scanned with OKEN Scanner

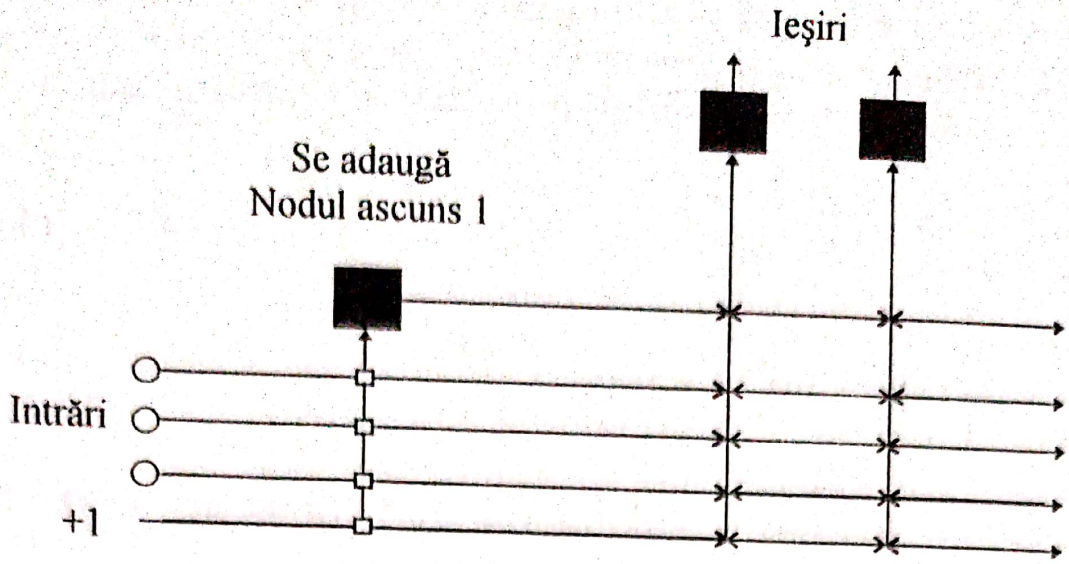


Fig. 4.6 Structura reţelei CASCOR după adăugarea unui nod ascuns. Activările se sumează pe liniile verticale. Conexiunile marcate □ sunt înghețate, conexiunile marcate X se modifică prin învăţare. Nodurile marcate ■ sunt noduri active (funcţie de activare *tanh*)

Se prezintă de câteva ori setul de date reţelei, modificând ponderile conexiunilor de intrare în nodul ascuns după fiecare trecere. Scopul este maximizarea modulului corelaţiei S_h dintre ieşirea noului nod introdus, $z_h(\xi)$, şi semnalul eroare $C_j(\xi)$ în nodul de ieşire j:

$$S_h = \sum_{j=1}^{N} \left| \sum_{\xi=1}^{P} \left(z_h(\xi) - \bar{z}\right)\left(C_j(\xi) - \bar{C}_j\right) \right| \qquad (4.74)$$

S-au notat cu $\bar{z}$, $\bar{C}_j$, valorile medii ale $z_h(\xi)$ şi $C_j(\xi)$, pe toate patternurile. In varianta propusă de Fahlman, se folosesc noduri ascunse şi de ieşire cu funcţie de activare sigmoidală simetrică, de tip tangentă hiperbolică.

Pentru maximizarea expresiei (4.74) se calculează derivata corelaţiei S_h în raport cu fiecare pondere de intrare în nodul candidat ($\partial S_h / \partial w_{ih}$) şi apoi, paşii sunt similari modului de obţinere a legii BKP. Funcţia eroare totală poate fi norma L_2 a erorii obţinute la ieşirea reţelei:

$$C = \frac{1}{2}\sum_{j=1}^{N}\sum_{\xi=1}^{P}\left[y_j(\xi) - d_j(\xi)\right]^2 = \frac{1}{2}\sum_{j=1}^{N}\sum_{\xi=1}^{P}\left[\left(\sum_{h=1}^{H} v_{hj}\, z_h\right) - d_j(\xi)\right]^2 \qquad (4.75)$$

cu $y_j(\xi)$ este valoarea obţinută şi $d_j(\xi)$ valoarea dorită la ieşirea j, fiecare pentru patternul ξ. S-a notat cu v_{hj} ponderea conexiunii dintre nodul ascuns h şi nodul de ieşire j. Minimizarea erorii C se realizează prin scăderea gradientului, calculând:

$$C_j(\xi) = [y_j(\xi) - d_j(\xi)]\, f'(\xi) \tag{4.76}$$

$$\frac{\partial C}{\partial v_{hj}} = \sum_{\xi=1}^{P} C_j(\xi)\, x_h(\xi) \tag{4.77}$$

cu $x_h(\xi)$ valoarea de intrare în noul nod h şi $f'(\xi)$ derivata de ordinul întâi a funcţiei de activare a nodului de ieşire, pentru patternul ξ.

Derivata corelaţiei în raport cu fiecare pondere este (4.78), unde s-a notat cu $\delta(\xi)$ expresia (4.79) şi cu sgn_j semnul corelaţiei dintre noul nod introdus şi eroarea în nodul de ieşire j.

$$\frac{\partial S_h}{\partial w_{ih}} = \sum_{\xi=1}^{P} \delta(\xi)\, x_i(\xi) = \sum_{\xi=1}^{P} \sum_{j=1}^{N} \operatorname{sgn}_j \left(C_j(\xi) - \overline{C}_j\right) f'(\xi)\, x_i(\xi) \tag{4.78}$$

$$\delta(\xi) = \sum_{j=1}^{N} \operatorname{sgn}_j \left(C_j(\xi) - \overline{C}_j\right) f'(\xi) \tag{4.79}$$

Pentru maximizarea valorii S_h, modificarea ponderilor conexiunilor de intrare în orice nod h, se poate realiza cu algoritmul standard BKP [R3]:

$$w_{ih}(k+1) = w_{ih}(k) + \eta \left(\frac{\partial S_h}{\partial w_{ih}}\right) = w_{ih}(k) + \eta \sum_{\xi=1}^{P} \delta(\xi)\, x_i(\xi) =$$

$$= w_{ih}(k) + \eta \sum_{\xi=1}^{P} \sum_{j=1}^{N} \operatorname{sgn}_j \left(C_j(\xi) - \overline{C}_j\right) f'(\xi)\, x_i(\xi) \tag{4.80}$$

cu η este rata de învăţare şi k indexul iteraţiei. Utilizarea BKP conduce însă la crearea unui număr mai mare de noduri ascunse, decât utilizarea algoritmului Quickprop, aplicat în varianta propusă de Fahlman [H1].

Se poate remarca, de asemenea, că $\partial C/\partial v = -\partial S/\partial w$.

Atunci când valoarea lui S nu se mai îmbunătăţeşte (nu mai creşte), înseamnă că un nou nod ascuns este într–adevăr adăugat în structură (fig. 4.7).

Scanned with OKEN Scanner

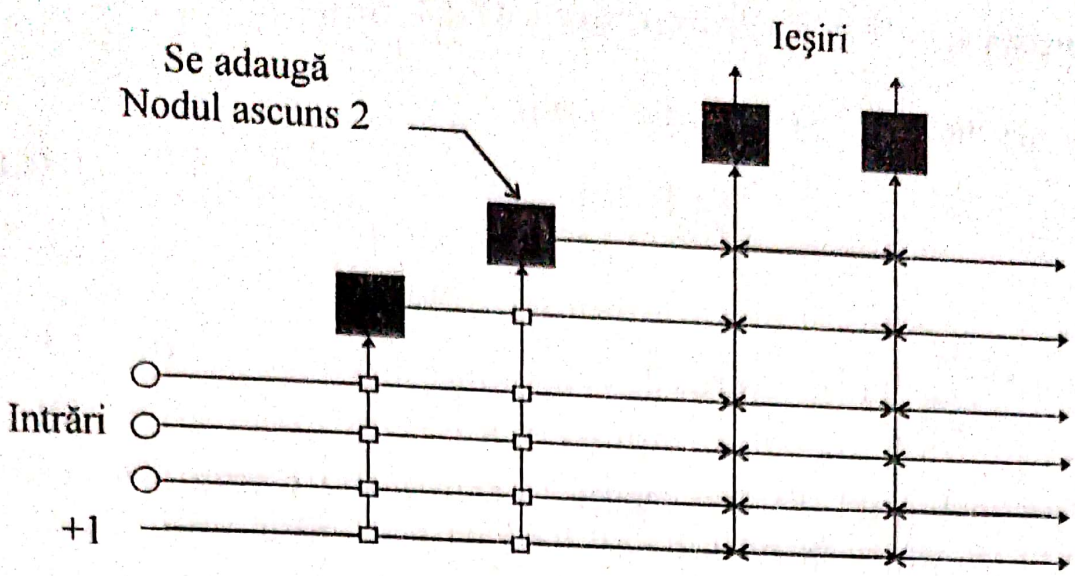


Fig. 4.7 Structura reţelei CASCOR după adăugarea a două noduri ascunse. Activările se sumează pe liniile verticale. Conexiunile marcate □ sunt înghețate, conexiunile marcate X se modifică prin învăţare. Nodurile marcate ■ sunt noduri active (funcţie de activare *tanh*)

Ponderile sale spre stratul de intrare sunt înghețate, iar ponderile tuturor conexiunilor spre stratul de ieşire (de la intrare – ieşire şi de la toate nodurile ascunse la ieşire) se modifică prin învăţare.

Succesiunea etapelor anterioare instalării definitive a nodului ascuns în structură poartă numele de "titularizare ("tenure") [H1]. Ciclul de adăugare a unui nod ascuns şi re–învăţare a stratului de ieşire continuă până eroarea scade sub un prag, sau proiectantul renunţă.

Datorită luării în considerare a valorii absolute în relaţia (4.74), este importantă doar valoarea modulului corelaţiei S, nu şi semnul său. Dacă $S > 0$, atunci valoarea ponderii conexiunii dintre nodul candidat şi cel de ieşire este negativă, pentru a prelua o parte a erorii; dacă însă corelaţia $S < 0$, atunci valoarea ponderii conexiunii dintre nodul candidat şi cel de ieşire este pozitivă.

Se poate folosi un ansamblu de noduri candidat, fiecare având un set inițial de ponderi diferit. Toate primesc aceleaşi semnale de intrare, utilizează aceeaşi eroare obţinută pentru fiecare pattern. Datorită faptului că nu interacţionează, nu vor afecta reţeaua în timpul învăţării, deci nodurile candidat pot învăţa în paralel. In final, va fi instalat candidatul cu valoarea cea mai bună a corelaţiei.

Pentru cazul în care funcţia ce trebuie aproximată de reţeaua CASCOR este o sumă de sigmoide cu număr finit de termeni, se demonstrează că, algoritmul converge cu o viteză de cel puţin ordinul $O(1 / H)$, cu H numărul nodurilor ascunse [D30].

Scanned with OKEN Scanner

Teorema 4.2 (Teorema de convergenţă a CASCOR)

Fie o funcţie $d : \Re^n \to \Re$, dată de relaţia:

$$d = \sum_{j=0}^{n^*} v_j^* z_j^* \tag{4.81}$$

cu

$$z_j^* = \tanh\left(\sum_{i=0}^{M} w_{ij}^* x_i\right), \quad 1 \le j \le n^*; \quad z_0^* = 1 \tag{4.82}$$

atunci algoritmul CASCOR este capabil să aproximeze funcţia d şi să obţină convergenţa cu viteza de ordinul $O(1 / H)$ cu H numărul final al nodurilor ascunse [D30].

Pentru demonstraţie, se consideră structura CASCOR simplificată (fig. 4.8) şi notaţiile:

$\mathbf{x} = [x_0, x_1, \ldots x_M]^T$	vectorul de intrare al reţelei
M	numărul nodurilor din stratul de intrare
$\mathbf{z}^k = [z_0, z_1, \ldots z_k]^T$	vectorul de ieşire din stratul ascuns, la iteraţia k
$x_0 = z_0 = 1$	valorile de prag
$\{w_{ij} \mid 1 \le i \le M, j = 1\}$	ponderile conexiunilor dintre stratul de intrare şi nodurile ascunse
$\mathbf{v}^k = \left[v_0^k, v_1^k, \ldots, v_k^k\right]^T$	ponderile dintre stratul ascuns şi perceptronul simplu de ieşire, la iteraţia k
$N = 1$	numărul nodurilor din stratul de ieşire

Componentele vectorului $\mathbf{v}^k$ se modifică atunci când se adaugă un nou percepron simplu în structură. Indicele k[16] indică faptul că vectorul $\mathbf{v}^k$ s-a modificat după inserarea nodului indice k (la iteraţia k).

Fiecare nod ascuns calculează valoarea de ieşire z_h conform unei relaţii similare expresiei (4.73), dar cu limitele de sumare $0 \le i \le M$. Valoarea de ieşire a perceptronului simplu din ultimul strat, la iteraţia k (după adăugarea neuronului ascuns indice k în structură), este:

$$y^k = \sum_{h=0}^{k} v_h^k z_h = \left(\mathbf{v}^k\right)^T \mathbf{z}^k \tag{4.83}$$

[16] Utilizarea indicelui k este justificată de faptul că, inserarea câte unui nou nod are loc la fiecare iteraţie k. In poziţia superscript, indicele marchează iteraţia (ex.: e^k în loc de $e(k)$; spre deosebire de restul lucrării, unde iteraţia / epoca a fost marcată prin plasarea indicelui în poziţie subscript), iar în poziţia subscript, indicele marchează componentele vectorului.

Scanned with OKEN Scanner

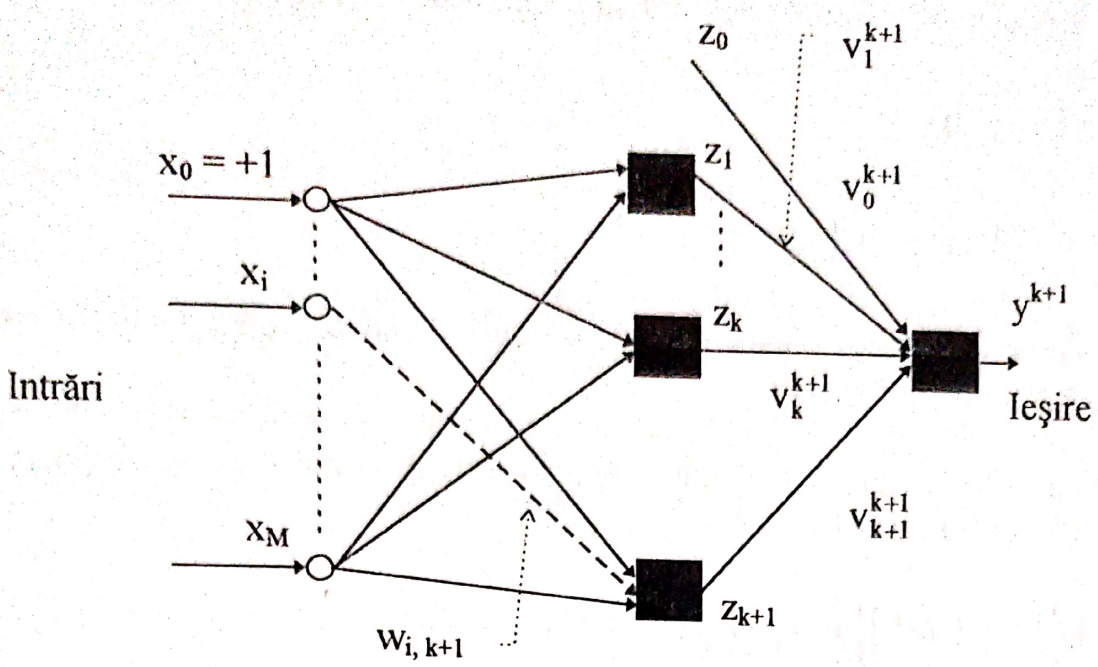


Fig. 4.8 Structura reţelei CASCOR, cu notaţiile utilizate

Eroarea la iteraţia k, atunci când numărul neuronilor ascunşi activi este k, este $e^k = d - y^k$, iar eroarea pătratică medie totală este $C^k = \| e^k \|^2$

Covariaţia nodului ascuns indice $k + 1$ este dată de relaţia (4.84), unde valorile medii $\bar{e}^k$, $\bar{z}_{k+1}$ sunt exprimate de (4.85).

$$r^{k+1} = E\left\{\left(e^k - \bar{e}^k\right)\left(z_{k+1} - \bar{z}_{k+1}\right)\right\} \tag{4.84}$$

$$\bar{e}^k = E\left\{e^k\right\} \text{ şi } \bar{z}_{k+1} = E\left\{z_{k+1}\right\} \tag{4.85}$$

Cei doi paşi de învăţare în structura CASCOR sunt:

❶ Se determină ponderile de la stratul de intrare la nodul ascuns, prin maximizarea covariaţiei:

$$| r^{k+1} | = \max_{\{w_{i,\,k+1},\, 0 \le i \le M\}} \left| E\left\{\left(e^k - \bar{e}^k\right)\left(z_{k+1} - \bar{z}_{k+1}\right)\right\} \right| \tag{4.86}$$

❷ Se determină ponderile dintre toate nodurile ascunse şi ieşire, precum şi dintre stratul de intrare şi ieşire, prin minimizarea erorii pătratice medii:

$$C^{k+1} = \min_{\Delta v^{k+1},\, v_{k+1}^{k+1}} \| e^{k+1} \|^2 \tag{4.87}$$

Valoarea v_0^0 trebuie determinată astfel încât C^0 să fie minimă:

$$v_0^0 = E\{d\} \tag{4.88}$$

$$C^0 = \| d \|^2 - \left(E\{d\}\right)^2 \text{ şi } \bar{e}^0 = 0 \tag{4.89}$$

Când valoarea C^k este mai mică decât o valoare predefinită, nu se mai adaugă alţi neuroni ascunşi.

Pentru demonstrarea teoremei, trebuie aflat minimul expresiei din ecuaţia (4.87), prin rezolvarea sistemului liniar (4.90), cu ajutorul caruia se poate arăta că este adevărată relaţia (4.91).

$$E\left\{\left(e^{k+1}\right)\left(z^k\right)\right\} = 0, \quad E\left\{\left(e^{k+1}\right)\left(z_{k+1}\right)\right\} = 0 \tag{4.90}$$

$$C^{k+1} = C^k - \frac{\left(r^{k+1}\right)^2}{\| z_{k+1} \|^2 - q^{k+1}} \le C^k - \frac{\left(r^{k+1}\right)^2}{\| z_{k+1} \|^2} \tag{4.91}$$

Intrucât $\| z_{k+1} \|^2 \le 1$, rezultă relaţia (4.92). S-au utilizat notaţiile (4.93), (4.94).

$$C^{k+1} \le C^k - \left(r^{k+1}\right)^2 \tag{4.92}$$

$$q^{k+1} = E\left\{\left(z_{k+1}\right)\left(z^k\right)^T\right\}\left(S^k\right)^{-1} E\left\{\left(z^k\right)\left(z_{k+1}\right)\right\} > 0 \tag{4.93}$$

$$S^k = E\left\{\left(z^k\right)\left(z^k\right)^T\right\} \tag{4.94}$$

Trebuie arătat că termenul $(r^{k+1})^2$ din expresia (4.92) nu este zero, pentru a arăta că $C^{k+1} < C^k$. Pentru aceasta, folosind relaţia (4.86) se obţine condiţia necesară:

$$\max_{1 \le j \le n^*} v_j^* \; E\left\{\left(e^k\right)\left(z_j^*\right)\right\} \ge \frac{C^k}{n^*} > 0 \tag{4.95}$$

Aceasta permite totdeauna, determinarea perceptronului simplu ascuns indice $k+1$, pentru care:

$$\left(r^{k+1}\right)^2 > \frac{\left(C^k\right)^2}{b_f^2} \tag{4.96}$$

unde s-a notat cu b_f expresia:

$$b_f = \left(n^*\right)^2 \left(\max_{1 \le j \le n^*} |v_j^*| \right)^2 \tag{4.97}$$

Se observă că, b_f^2 depinde numai de funcţia d, nu şi de numărul iteraţiei, k [D30]. Din relaţiile (4.89), (4.92) şi (4.96) rezultă:

$$C^{k+1} < \frac{b_f^2 C^0}{(k+1) C^0 + b_f^2} \tag{4.98}$$

care arată convergenţa de ordinul $O(1 / k)$ şi q.e.d [D30].

Metoda CASCOR a fost aplicată pentru rezolvarea problemei "două spirale", propusă de Alexis Wieland de la MITRE Corporation, precum şi pentru problema parităţii n – dimensionale. In primul caz rezultă reţele cu un număr de 12 până la 19 noduri ascunse, cu media 15.2 şi valoarea mediană 15. Viteza de convergenţă măsurată în număr de epoci, este de 5 ori mai mare decât în cazul utilizării algoritmului Quickprop şi de 10 ori mai mare decât pentru BKP, ambele pentru structuri fixe cu 15 noduri ascunse. Avantajul din punct de vedere al vitezei se datorează pe de o parte faptului că, în cazul metodei CASCOR este necesară numai trecerea "înainte" prin reţea a datelor, fără propagarea inversă a erorii. De asemenea, în CASCOR, timp de multe epoci de învăţare, reţeaua are dimensiuni mult mai reduse (la începutul aplicării algoritmului) decât cele finale.

Pentru problema parităţii 8 – dimensionale, rezultate comparative sunt prezentate în tab. 4.8 [F18].

Tab. 4.8 CASCOR vs. Quickprop vs. BKP în problema parităţii 8 – dimensionale

Metoda	Nr. noduri ascunse	Nr. epoci necesare convergenţei
CASCOR	4 – 5	357
Quickprop	16	86
BKP	16	2000

Scanned with OKEN Scanner

Alte aplicaţii ale metodei sunt: segmentarea imaginilor creierului obţinute prin rezonanţă magnetică, clasificarea celulelor măduvei spinării, implementarea transformărilor cinematice inverse ale braţului unui robot, modelarea procesului cognitiv uman şi modelarea seriilor de timp haotice. Există, de asemenea, aplicaţii ale unor variante ale metodei CASCOR, care iau în considerare conectarea fiecărui nod de ieşire la propriul set de noduri ascunse, sau combinarea structurii cascadă cu noduri ascunse având funcţii de activare performante.

Printre avantajele metodei se numără:

❶ Nu este necesară stabilirea apriori a dimensiunilor şi conectivităţii reţelei. Rezultă o reţea de dimensiuni rezonabile (deşi nu optimală).

❷ Invăţarea este rapidă. O lege empirică ce aproximează variaţia timpului de învăţare măsurat în epoci, în funcţie de numărul H al nodurilor ascunse rezultate în final, pentru rezolvarea unei probleme, este $H \log H$ [F18].

❸ In orice moment, învaţă un singur strat de ponderi, restul reţelei nefiind modificat.

❹ Nu este necesară propagarea inversă a erorii prin reţea. Un singur semnal de eroare poate fi trimis tuturor candidaţilor.

❺ Când se folosesc ansambluri de noduri candidat, acestea nu interacţionează între ele, cu excepţia etapei de alegere a nodului candidat câştigător. Fiecare nod primeşte aceleaşi valori de intrare şi semnale de eroare, în acelaşi timp cu celelalte noduri. Această comunicaţie limitată face structura atractivă din punct de vedere al implementării paralele.

❻ Software de tip "public domain" pentru CASCOR a devenit disponibil din anul 1990, odată cu rapoartele lui Scott Fahlman. Comunitatea de utilizatori CASCOR a crescut de atunci, dar numărul acestora este încă relativ redus în comparaţie cu cel al utilizatorilor algoritmului BKP.

Dezavantajele metodei sunt:

❶ Poate genera reţele cu număr mare de straturi ascunse, ceea ce sugerează investigarea posibilităţilor de oprire a creşterii în adâncime a structurii.

❷ Intrucât se adaugă noduri ascunse care sunt total interconectate cu toate intrările şi cu toate nodurile ascunse inserate anterior în reţea, numărul ponderilor creşte rapid şi este evident pericolul apariţiei fenomenului de supraînvăţare [H29].

❸ Ca şi celelalte metode constructive, nu este optimală.

❒ *Metoda CL ("Constructive Learning")* (II)

Metoda de "învăţare constructivă" CL ("Constructive Learning") se bazează pe adăugarea unui nou nod ascuns atunci când eşantionul de intrare nu este învăţat corect [H30]. Noul neuron ascuns va avea valori ale ponderilor care

să facă posibilă activarea sa de către eşantionul de intrare curent, şi inactivarea sa de către toate eşantioanele anterioare.

❒ ***Algoritmul LPPA ("Linear – Programming Perceptron Algorithm") (III)***

Metoda LPPA ("Linear – Programming Perceptron Algorithm"), propusă de Frattale Mascioli, Martinelli, ş.a. [M59] construieşte structuri piramidale sau cascadă, având la bază perceptroni simpli cu funcţii de activare de tip treaptă. Structura cascadă poate fi transformată într–un perceptron cu două straturi active, prin rearanjarea nodurilor într–un strat de intrare în perceptron. Cel de–al doilea strat activ (strat de ieşire) conţine un singur perceptron simplu. Algoritmul nu necesită setarea anumitor parametri sau stabilirea criteriului de oprire a învăţării.

4.5.5.2 Metode ce interconectează parţial elementele introduse în reţea, cu cele existente

❒ ***Metoda GMDH ("Group Method of Data Handling") (I)***

Metoda GMDH ("Group Method of Data Handling") a fost propusă de A.G. Ivakhnenko în anul 1968 [K33], şi se bazează pe un algoritm ce generează structuri cu mai multe straturi ascunse. Modul de introducere a nodurilor ascunse în reţea nu este predeterminat. Dintr-un lot de candidaţi conectaţi diferit în structură, se alege în mod competitiv un câştigător, care se interconectează parţial cu nodurile deja existente. Semnalul de ieşire din fiecare nod se exprimă ca un polinom, depinzând de semnalele de intrare în reţea.

O reţea tipică GMDH, pentru cazul în care $M = 4$ intrări, este prezentată în fig. 4.9. Fiecare nod primeşte două semnale de intrare. Există, de asemenea, mai multe straturi de elemente de procesare, iar ultimul dintre acestea conţine un singur nod, fiind precedat de un strat ascuns cu două noduri. Construirea structurii se realizează prin adăugarea câte unui strat. Semnalul de ieşire al oricărui nod se poate scrie ca o combinaţie polinomială a semnalelor de intrare. Fiecare strat nou introdus creşte gradul polinomului cu 2, astfel încât semnalul de ieşire din reţea se scrie ca un polinom de gradul $2L$, cu L numărul total al straturilor (excluzând stratul de intrare), numit polinomul lui Ivakhnenko.

Algoritmul GMDH a fost aplicat iniţial pentru predicţia populaţiei piscicole în mări şi oceane, iar ulterior pentru rezolvarea problemelor de regresie.

Dezavantajele metodei sunt:

❶ Număr mare de operaţii matriciale, deci metoda este consumatoare de timp.

Scanned with OKEN Scanner

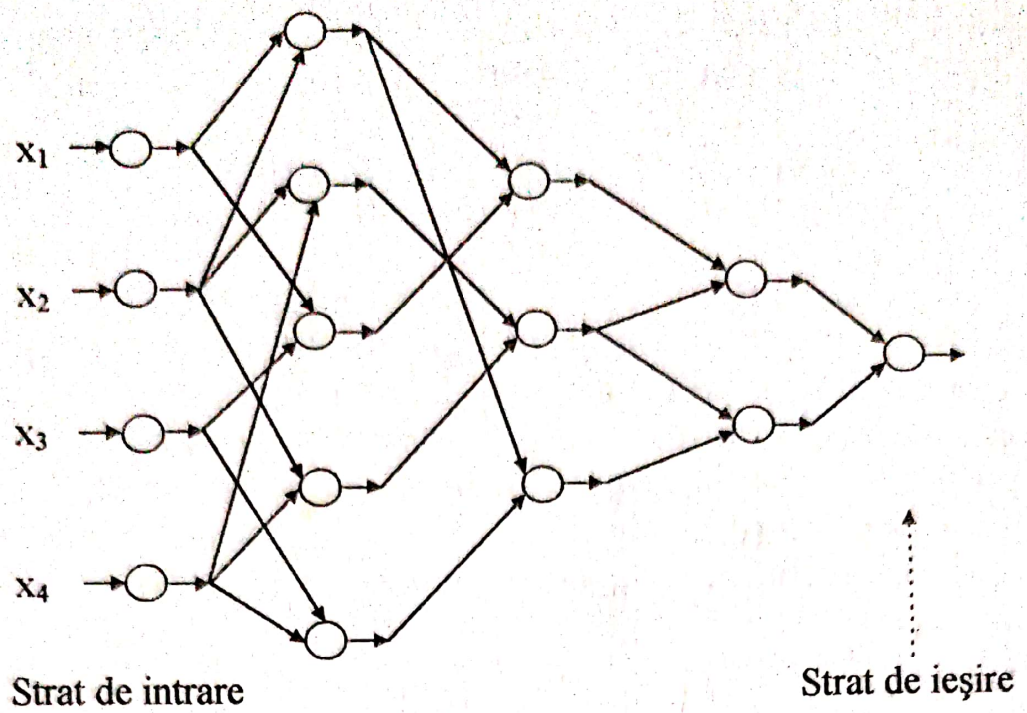


Fig. 4.9 Reţea GMDH

❷ Coeficienţii polinomului ce descrie reţeaua nu au o interpretare directă, ca ponderile conexiunilor în alte metode.

❸ Este necesar un volum mare de date de învăţare, de forma (x_ξ, d_ξ).

4.5.6 Metode care generează arbori

❒ ***Algoritmul Upstart* (I)**

Algoritmul Upstart [F502] construieşte reţeaua în timpul învăţării, prin adăugarea unor noi perceptroni simpli, în scopul scăderii erorii. Nodurile sunt neuroni binari iar structura rezultată este o piramidă (arbore) cu ieşirea reţelei în vârf (fig. 4.10). Fiecare neuron este conectat cu alţi doi neuroni, într–o structură "feedforward", construită astfel: pentru un singur neuron de ieşire, se urmăreşte clasificarea realizată (valori 0 sau 1) şi se notează toate cazurile eronate. Apoi, se adaugă nodurile notate cu ②, unul pentru a corecta situaţiile greşite pentru valoarea 1, iar celălalt pentru valoarea 0. Acestea sunt conectate la nodurile de ieşire prin ponderi cu valori mari pozitive şi respectiv, negative, astfel încât efectul ieşirii din etapa precedentă devine neglijabil. Nodurile sunt de asemenea, conectate direct la intrare. Dacă este necesar, se mai adaugă alţi perceptroni simpli notaţi cu ③, pentru a corecta greşelile nodurilor abia adăugate, etc.

Scanned with OKEN Scanner

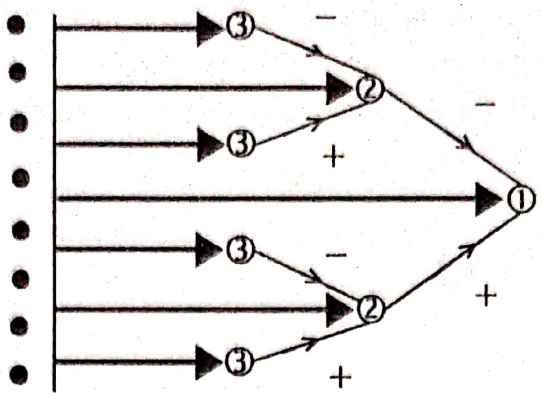


Fig. 4.10 Construirea structurii neurale, pe baza algoritmului Upstart. Nodurile sunt numerotate în ordinea introducerii lor, iar săgeţile reprezintă conexiuni de la toate nodurile de intrare aflate la baza săgeţii

Deşi structura rezultată este ierarhică şi cu un aspect diferit de cele obţinute prin alte metode, aceasta poate fi rearanjată sub forma unei RPM cu un singur strat ascuns. Toate nodurile sunt plasate în stratul ascuns, apoi se adaugă un nou nod de ieşire, total interconectat cu acesta. Stabilirea corespunzătoare a ponderilor determină obţinerea performanţelor identice cu cele ale reţelei – arbore.

Algoritmul Upstart se aplică numai pentru probleme de clasificare.

❒ ***Metodă care generează un arbore*** **(II)**

Metoda propusă de Nahban şi Zomaya [N12] este o metodă incrementală, care generează în final un arbore. Având în vedere faptul că, operaţiile efectuate sunt "crează / şterge element", unde elementul este neuron / strat, această metodă poate fi inclusă şi în categoria metodelor constructive – "distructive". Cu toate acestea, întrucât structura iniţială este mai redusă decât structura finală, o vom prezenta pe scurt în continuare.

Structura luată în considerare este o RPM, iar algoritmul este BKP standard, bazat pe metoda gradientului negativ. Deci, se urmăreşte îndeplinirea condiţiei $||\nabla C|| = 0$, unde:

$$\| \nabla_w E \| = \| \nabla_w C \| = \sqrt{\sum_{i \in \mathbf{C}} (\Delta w_i)^2} \qquad (4.99)$$

S-a notat cu $\mathcal{C}$ setul tuturor conexiunilor din reţea.

Ideea algoritmului este aceea de a evalua performanţa reţelei în timpul învăţării şi de a genera structuri mai "promiţătoare", dacă performanţa se modifică în sens nedorit (tab. 4.9). Se consideră drept criteriu de comparaţie, valoarea funcţiei:

$$\| \nabla_w C_{av} \| = \sqrt{\sum_i (\Delta w_i)^2_{av}} \tag{4.100}$$

$$(\Delta w_i)_{av} = \frac{\sum_\xi (\Delta w_i)_\xi}{P} \tag{4.101}$$

S–a notat cu $(\Delta w_i)_\xi$ modificarea ponderii conexiunii i, când la intrare se prezintă patternul ξ. In acest fel, performanţa reţelei nu mai depinde dimensiunea setului de date de învăţare.

Generarea unei noi structuri se face prin crearea mai multor variante modificate ale reţelei iniţiale şi selectarea celei mai bune dintre acestea. Etapa de generare se desfăşoară în două etape: faza de eliminare şi cea de adăugare. Mai întâi se elimină un neuron sau un strat. Dacă structura rezultată este satisfăcătoare, atunci este acceptată. Altfel, se execută faza de adăugare: se introduce un neuron / un strat în structură şi rezultă două variante. Se alege cea mai bună variantă.

La introducerea unui nou strat, între ultimul strat ascuns şi cel de ieşire, se consideră relaţia empirică:

$$H_{nou} = \left[\frac{N + H_{ultim}}{2} \right] \tag{4.102}$$

unde H_{nou} şi H_{ultim} sunt numerele de noduri ascunse din noul strat, respectiv din ultimul strat ascuns existent anterior.

Tab. 4.9 Generarea structurilor "promiţătoare"

Start cu o structură de dimensiuni mici.
Repetă
Calculează $\| \nabla_w C_{av} \|$ după fiecare iteraţie k.
Dacă $\| \nabla_w C_{av} \|_k \geq \| \nabla_w C_{av} \|_{k-1}$, structura nu este viabilă.
Dacă structura nu este viabilă, atunci generează o altă structură.
Continuă procesul de învăţare.
Până la convergenţă.

Scanned with OKEN Scanner

Tab. 4.10 Crearea unei noi structuri

Crează o nouă structură:
Dacă noua structură nu este viabilă sau este inutilă
atunci prezintă inventarul tuturor structurilor inutile.
Modifică vechea structură.
Faza de somn

Repetă
Execută o iteraţie de învăţare cu somn
Dacă $C_{\text{noua structură}} \leq C_{\text{vechea structură}}$ atunci succes şi stop.
Dacă $C_k \geq C_{k-1}$ în faza de somn, atunci stop şi eşec.
Până se execută numărul maxim de iteraţii de somn.
□
Dacă s–a atins numărul maxim de iteraţii de somn, atunci
structura nu este viabilă. Ieşire din faza de somn.
Dacă eşec, atunci structura obţinută nu este promiţătoare.
Altfel evaluează noua structură.

Parantezele semnifică "parte întreagă din". Intrucât această relaţie este empirică, metoda poate face parte şi din categoria metodelor empirice (cap. 4.3).

Crearea unei noi structuri se realizează conform tab. 4.10. Noua structură este promiţătoare dacă $\| \nabla_w C_{av} \|$ descreşte sau se obţine o descreştere semnificativă a erorii faţă de structura anterioară; altfel spus, dacă este valabilă relaţia (4.103) sau, dacă este îndeplinită condiţia (4.104), unde $K-1$ este prima iteraţie după faza de somn, DR este rata de descreştere a valorii erorii.

$$\| \nabla_w C_{av} \|_k \leq \| \nabla_w C_{av} \|_{k-1} \qquad (4.103)$$

$$(C_{\text{vechea structură}} - C_{\text{noua structură}}) \geq (DR \times C_{\text{vechea structură}}),\ 0 < DR < 1 \qquad (4.104)$$

Metoda este o variantă a algoritmului general "generează şi testează" şi conţine o serie de alte teste de eliminare a structurilor neviabile sau inutile, precum şi de revenire în noduri anterioare ale arborelui, în cazul atingerii unui punct mort ("dead end").

Metoda este evaluată pentru: ① aproximarea funcţiilor polinomiale de tipul $x^n + y^n$, unde $x, y \in [-2, +2]$, structura iniţială fiind 2 : 2 : 1; ②aproximarea funcţiei sinusoidale $sin(a\ x)$; ③ probleme de control. In primul caz, după 5000 de iteraţii, $DR = 0.5$, 200 cicluri de somn, se obţin rezultatele din tab. 4.11. Nahban şi Zomaya [N12] prezintă arborii care rezultă pentru fiecare caz în parte.

Scanned with OKEN Scanner

De exemplu, pentru funcţia $x^5 + y^5$ rezultă arborele din fig. 4.11, unde nodul haşurat semnifică nod inutil ("dead end"), din care trebuie realizată o întoarcere spre un nod precedent al arborelui.

Un avantaj esenţial al metodei este faptul că, algoritmul generează structuri a căror dimensiune este proporţională cu complexitatea problemei. Pe de altă parte, metoda nu conduce la cele mai simple structuri, existând cazuri în care dimensiunea obţinută nu mai este rezonabilă (ultima linie a tabelului).

Tab. 4.11 Rezultatele metodei, în cazul aproximării funcţiilor polinomiale

Funcţia	Structura finală	Nr. ponderi
x^2+y^2	2 : 3 : 1	13
x^3+y^3	2 : 4 : 2 : 1	25
x^4+y^4	2 : 6 : 1	25
x^5+y^5	2 : 6 : 3 : 2 : 1	50
x^6+y^6	2 : 6 : 3 : 1	43
x^7+y^7	2 : 8 : 4 : 2 : 1	73

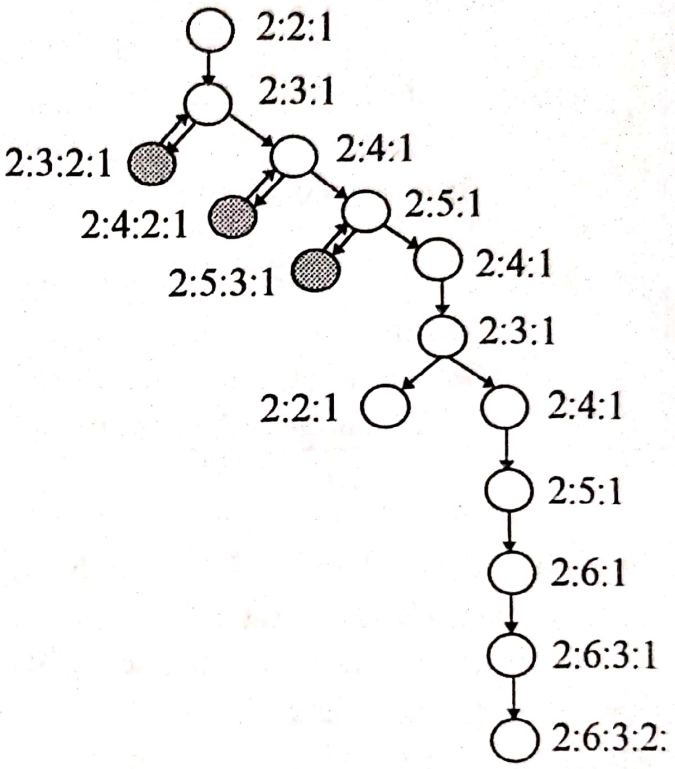


Fig. 4.11 Arborele generat în cazul aplicării metodei pentru aproximarea funcţiei $x^5 + y^5$

4.5.7 Metode care construiesc reţeaua pornind de la un arbore

❒ ***Metoda reţelelor entropice* (I)**

Sethi [S40] a propus construirea reţelelor neurale numite "entropice", pe baza arborilor de decizie. Aceştia se mai numesc şi "clasificatori ierarhici" şi au, ca şi reţelele neurale, capacitatea de a genera zone de decizie complexe în spaţiul n – dimensional.

Există însă şi deosebiri între clasificatorii cu arbori de decizie şi reţelele neurale: ① arborii sunt secvenţiali, în timp ce reţelele neurale multistrat au un

caracter puternic paralel; ② arborii au o capacitate de generalizare redusă, faţă de reţelele neurale.

Un arbore de decizie este caracterizat de un set de noduri, aşezate într-o anumită ordine. Fiecărui nod intern îi este asociată o funcţie de decizie, cu unul sau mai multe argumente. Arborele realizează o împărţire ierarhică a spaţiului de decizie. Pornind de la nodul rădăcină, fiecare partiţie realizată de nodurile interne succesive, presupune împărţirea regiunii de decizie în două spaţii. Funcţia de decizie a nodului defineşte hiperplanul ce le desparte. Clasificarea pe baza arborelui de decizie se face prin traversarea acestuia de la nodul – rădăcină spre un nod – frunză.

Pe de altă parte, fiind dat un arbore de decizie, este posibilă restructurarea acestuia sub forma unei RPM care este numită "reţea entropică". Regulile pe baza cărora se face translatarea sunt:

❶ Numărul de perceptroni simpli din primul strat al RPM este egal cu numărul de noduri interne ale arborelui de decizie. Fiecare dintre neuroni implementează o funcţie de decizie a unui nod intern.

❷ Toate nodurile – frunză corespund unui neuron din al doilea strat ascuns care, pentru arborii binari, realizează funcţia logică ŞI.

❸ Numărul neuronilor din stratul de ieşire este egal cu numărul claselor. Acest strat implementează, în cazul arborilor binari, funcţia logică SAU.

❹ Conexiunile dintre neuronii din stratul de intrare şi cei din primul strat ascuns realizează structura ierarhică a arborelui.

Pentru realizarea arborelui iniţial se poate folosi o metodă bazată pe calculul entropiei. Translaţia se realizează fără a se încerca să se optimizeze numărul nodurilor din stratul de intrare. Consecinţa translaţiei arbore – reţea este aceea că, se defineşte exact numărul de perceptroni simpli necesar în fiecare strat al reţelei. Paşii propuşi de Sethi pentru realizarea reţelei entropice sunt prezentaţi în tab. 4.12. Un exemplu de aplicare a metodei este acela al rezolvării problemei SAU – EXCLUSIV în variantă analogică: trebuie clasificate patternurile din fig. 4.12. Arborele de decizie este prezentat în fig. 4.13, în care nodurile de culoare diferită semnifică cele două clase distincte. Reţeaua entropică rezultată prin translaţia arbore – reţea este prezentată în fig. 4.14, unde culorile diferite ale nodurilor reprezintă clasele corespunzătoare.

Prin aplicarea metodei rezultă structuri cu număr redus de conexiuni, de aceea implementarea prin tehnici VLSI se simplifică.

Printre dezavantaje, se pot menţiona:

❶ Translaţia arbore – reţea neurală nu se realizează fără erori. Datorită faptului că fiecare nod al arborelui reprezintă o singură trăsătură, se poate întâmpla să rezulte arbori de dimensiuni foarte mari.

❷ Reţeaua fiind skeleton, nu există noduri redundante pentru absorbţia erorii, deci generalizarea nu este neaparat mai bună decât a unei structuri mari.

Scanned with OKEN Scanner

Tab. 4.12 Realizarea reţelei entropice

❶ Se împarte setul de date disponibil, compus din perechi intrare – ieşire dorită, în două părţi: setul de proiectare a arborelui şi setul de învăţare a reţelei. Dacă setul de date nu este suficient de mare, atunci se poate folosi în întregime, în fiecare dintre cele două etape menţionate. ❷ Se aplică un algoritm de realizare a arborelui pentru problema dată. ❸ Se realizează translatarea arbore → reţea neurală, pe baza regulilor precizate. ❹ Se asociază o funcţie sigmoidă fiecărui neuron. Se învaţă reţeaua folosind o lege de modificare a ponderilor de tip Widrow – Hoff.

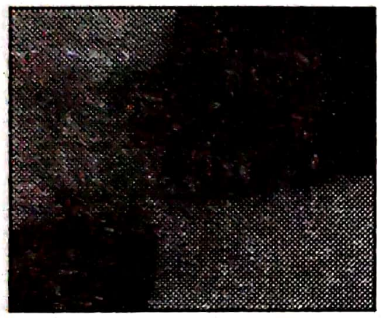


Fig. 4.12 Varianta analogică a problemei SAU – EXCLUSIV

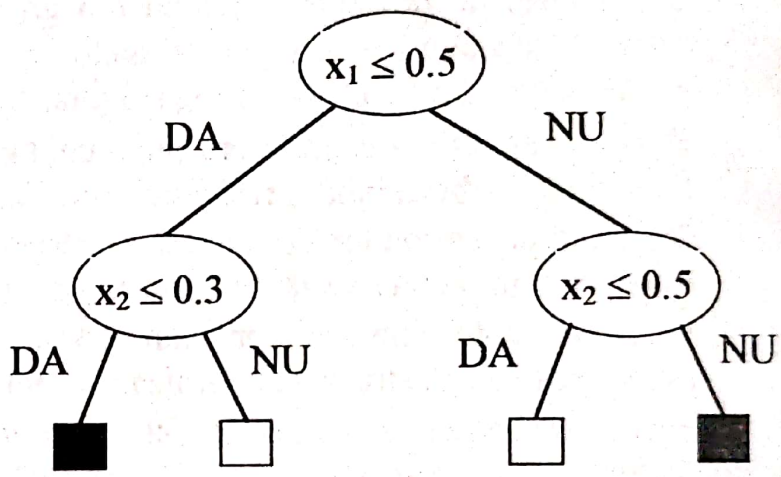


Fig. 4.13 Arborele generat pentru problema analogică SAU – EXCLUSIV

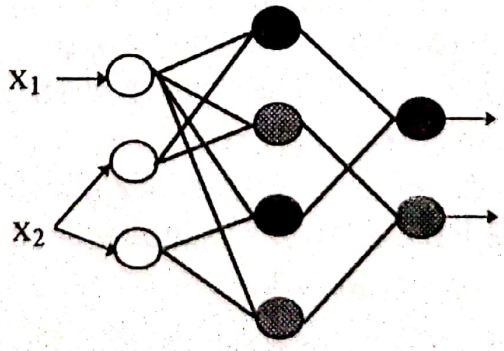


Fig. 4.14 Reţeaua entropică rezultată prin translaţia arborelui din fig. 4.13 în reţea

Scanned with OKEN Scanner

❸ Rețeaua skeleton se construiește pe baza cunoștințelor expert, care pot fi uneori incomplete. Pentru depășirea aceste situații, precum și a observației precedente, se propune o modificare în sensul creșterii rețelei minimale inițiale. Trebuie remarcat faptul că, metoda propusă de Sethi nu face parte propriu – zis din categoria algoritmilor constructivi, dar cu completarea propusă se poate considera că aparține acesteia [Y9].

Se propune învățarea rețelei skeleton, după ce au fost adăugate noduri redundante și se arată că generalizarea se îmbunătățește. Motivația este aceea că, la propagarea inversă a erorii, dacă în stratul spre care aceasta se propagă există prea puțini perceptroni simpli, nu-și vor putea modifica ponderile suficient, pentru a "absorbi" întreaga eroare propagată. Rețeaua skeleton (rețeaua entropică inițială) este denumită rețea cu subînvățare, iar procesul de adăugare a nodurilor redundante se numește proces de "potrivire" ("fitting").

❒ ***Metodă care construiește rețeaua pornind de la un arbore* (II)**

Ideea construirii unei rețele neurale pe baza arborelui de decizie stă la baza abordării realizate de Brent [B16]. Structura neurală are două straturi ascunse, cu N numărul nodurilor de ieșire, egal cu numărul claselor din problema de clasificare considerată.

Pentru construirea arborelui, se definește entropia arborelui T:

$$\mathcal{H}(\mathrm{T}) = \log\left(\frac{\mathrm{P}!}{\mathrm{P}_1!\ \mathrm{P}_2!\ldots\mathrm{P}_\mathrm{N}!}\right) = \log \mathrm{P}! - \sum_{\mathrm{n}=1}^{\mathrm{N}} \log \mathrm{P}_\mathrm{n}! \qquad (4.105)$$

Expresia din paranteze reprezintă numărul total de moduri în care, cele P valori ale setului de învățare, pot fi împărțite în clase marcate 1,...,N, cu constrângerea că, există P_1 valori marcate cu eticheta 1, P_2 valori marcate cu 2, etc.

Pentru $1 \le \xi \le P$, scriind $p_\xi = P_\xi / P$, se obține expresia (4.106), care sugerează criteriul entropic de realizare a arborelui.

$$\frac{\mathcal{H}(\mathrm{T})}{\mathrm{P}} = -\sum_{\mathrm{n}=1}^{\mathrm{N}} \mathrm{p}_\mathrm{n} \log \mathrm{p}_\mathrm{n} + O\left(\frac{\log \mathrm{P}}{\mathrm{P}}\right) \qquad (4.106)$$

Problema de studiu este clasificarea vocalelor din limba engleză britanică și australiană, iar performanțele, măsurate în procentajul erorilor pe setul de test, sunt aproximativ egale cu cele ale rețelei învățate cu algoritmul standard BKP. Viteza obținută este însă mai ridicată.

Scanned with OKEN Scanner

4.5.8 Metode de modularizare a rețelei

Pentru rezolvarea problemelor de dimensiuni mari, uneori este necesară împărțirea în subprobleme. Metodele de modularizare a structurii au la bază ideea împărțirii RPM în subrețele, fiecare capabilă să rezolve o parte a problemei date. Combinarea subrețelelor conduce la structura monolitică de rețea "feedforward" perceptron multistrat.

Aplicarea modularizării este utilă mai ales atunci când, o funcție complexă poate fi descompusă în funcții mai simple. Avantajele sunt legate mai ales de creșterea vitezei, a generalizării, precum și posibilitatea satisfacerii unor constrângeri impuse de hardware.

❒ ***Metodă de modularizare a rețelei* (I)**

O soluție de modularizare este secționarea stratului de ieșire. In cazul problemelor de clasificare, perceptronii simpli de ieșire reprezintă mai multe clase și partajarea se poate face astfel încât ieșirea unei subrețele să fie un subset al setului total de ieșiri. In cazul extrem, se poate alege varianta numită "o clasă – o rețea" ("one – class – one – net structure") [H30].

❒ ***Metode de modularizare a rețelei* (II)**

Hanson și Pratt [H500], Mirghafori, Morgan, ș.a. [M65], au aplicat modularizarea rețelei pentru clasificarea vocalelor. Se constată că, deși numărul aplicațiilor neurale modulare este mai mare în domeniul prelucrării semnalului vocal, există rețele modulare pentru control, precum și sisteme hibride de acest tip.

Invățarea subrețelelor este mai rapidă decât cea a rețelei monolitice. Din punct de vedere al performanțelor, nu se constată diferențe semnificative. Un alt avantaj este cel al comunicației rapide între subrețele (dacă este necesar), evident în cazul implementărilor paralele.

Printre dezavantajele metodei se pot menționa costul ridicat, datorat numărului mare de conexiuni (deci, și de ponderi din structură), care determină creșterea totală a dimensiunilor sistemului ce rezolvă problema dată.

❒ ***Metoda TADCM ("TAsk Decomposition through Competition in a Modular connectionist architecture")* (III)**

Jacobs [J4] a propus o structură modulară care se bazează pe descompunerea structurii în submodule, acestea învățând patternurile în urma unei competiții. Rezultatul acestei competiții este acela că, subrețele diferite învață patternuri de învățare diferite deci, învață funcții diferite. Structura

Scanned with OKEN Scanner

realizează descompunerea problemei globale în două sau mai multe subprobleme independente şi alocă subreţele neurale distincte pentru realizarea fiecăreia dintre acestea. In plus, metoda alocă fiecărei subprobleme structura neurală a cărei topologie este mai potrivită şi tinde să utilizeze aceeaşi reţea pentru subprobleme similare.

Metoda a fost testată pentru aplicaţii de recunoaşterea obiectelor şi localizare spaţială din imagini de retină simulate, precum şi în controlul simulat al braţului unui robot .

- ***Metoda TACOMA ("TAsk decomposition by COrrelation MeAsures")* (IV)**

Metoda TACOMA, propusă de Lange, Voigt, ş.a. [L39] realizează modificarea structurii (inspirată din metoda CASCOR) şi descompunerea în subprobleme.

Structura iniţială este compusă din noduri de intrare şi noduri de ieşire. Straturile ascunse sunt inserate unul câte unul, până când eroarea la ieşire scade sub o valoare prestabilită. Ponderile conexiunilor care intră în nodurile de ieşire sunt modificate în mod repetat, după inserarea fiecărui strat ascuns, în scopul minimizării erorii pătratice medii la ieşire. Există conexiuni intrare – ieşire şi, de asemenea, de la fiecare strat ascuns la ieşire. La fiecare iteraţie, numărul intrărilor în fiecare nod de ieşire creşte cu numărul nodurilor din noul strat ascuns introdus. Nodurile de ieşire sunt caracterizate de funcţii de activare sigmoidale, iar modificarea ponderilor menţionate se face cu algoritmul Quickprop [L39].

La introducerea unui nou strat ascuns, se determină numărul nodurilor acestuia printr-o procedură care identifică maximele proiecţiei erorilor de ieşire în spaţiul de intrare. Numărul maximelor corespunde numărului nodurilor din stratul ascuns. Ieşirea fiecărui nod ascuns este ponderată de o funcţie fereastră gaussiană, centrată pe un maxim în spaţiul de intrare. Proiectarea erorilor de ieşire în spaţiul de intrare se face printr-o metodă similară algoritmului Kohonen, care însă nu foloseşte relaţii de vecinătate.

Etapa de estimare a valorilor ponderilor conexiunilor dintre nodurile inserate anterior şi fiecare nou nod se numeşte "routare". Aceasta se realizează astfel încât să se maximizeze corelaţia dintre ieşirea sa şi eroarea la ieşirile nodurilor de ieşire, precum şi de a maximiza corelaţia cu semn schimbat, dintre toate nodurile stratului ascuns.

Rezultatele metodei TACOMA au fost evaluate pentru problema "două spirale", precum şi pentru "două spirale gemene". In primul caz, metoda TACOMA conduce la rezultate mai bune faţă de CASCOR. In cel de-al doilea caz, diferenţa este evidentă în sensul că TACOMA reuşeşte să clasifice corect, în timp ce rezultatul metodei CASCOR nu este satisfăcător.

Scanned with OKEN Scanner

Performanţele în probleme de clasificare dificile sunt mai bune decât cele ale metodei CASCOR. Un dezavantaj al metodei este numărul mare de extreme locale pe care le prezintă funcţia cost ce trebuie minimizată pentru realizarea optimizării. Iniţializarea funcţiei fereastră care ponderează ieşirea fiecărui nod ascuns influenţează rezultatul optimizării.

4.5.9 Alte metode constructive pentru structuri neurale "feeforward"

❒ *Alte metode constructive* (I)

Wang, ş.a. [W20], au propus utilizarea a trei criterii pentru dimensionarea celor două straturi ascunse ale unei structuri neurale "feedforward". Se folosesc notaţiile:

- **v** vectorul ieşirilor perceptronilor simpli din primul strat ascuns
- **V** matricea tuturor vectorilor **v**
- **z** vectorul ieşirilor perceptronilor simpli din cel de-al doilea strat ascuns
- **Z** matricea tuturor vectorilor **z**

iar restul simbolurilor au semnificaţia din par. 3.2.1.

Definiţia 4.3 Fie Λ un spaţiu vectorial n – dimensional şi $\theta_1, \theta_2, ..., \theta_r$ vectori în spaţiul Λ. Fiind dată matricea $\Theta = [\theta_1\ \theta_2 ...\ \theta_r]$ de dimensiuni n x r, grupul de vectori $\theta_1, \theta_2, ..., \theta_r$ se spune că este δ – liniar independent dacă şi numai dacă, determinantul matricii Θ satisface relaţia (4.107), unde δ este oricare număr real pozitiv de valoare mică.

$$\det(\Theta^T \Theta) > \delta \tag{4.107}$$

Dacă se defineşte dimensiunea spaţiului $\{V\}$ al ieşirilor nodurilor din primul strat ascuns, ca numărul maxim de vectori **v** care sunt δ – liniar independenţi, atunci matricea $\left(V^T V\right)_{H_1}$ este matrice δ – nesingulară, iar matricea $\left(V^T V\right)_{H_1+1}$ este δ – singulară, adică:

$$\det\left(\mathbf{V}^T \mathbf{V}\right)_{H_1} \geq \delta \quad \text{când} \quad \det\left(\mathbf{V}^T \mathbf{V}\right)_{H_1+1} < \delta \tag{4.108}$$

Valorile H_1 şi $H_1 + 1$ indică dimensiunea matricilor V şi V^T, în fiecare caz. Matricea $(V^T V)$ este o matrice pătrată, definită ca matricea de corelaţie pentru V.

Scanned with OKEN Scanner

Similar, dimensiunea spaţiului $\{\boldsymbol{Z}\}$ al ieşirilor nodurilor din cel de-al doilea strat ascuns este definită ca numărul maxim de vectori $\boldsymbol{z}$ care sunt δ – liniar independenţi:

$$\det\left(\mathbf{Z}^{\mathrm{T}}\,\mathbf{Z}\right)_{\mathrm{H}_2} \geq \delta \quad \text{când} \quad \det\left(\mathbf{Z}^{\mathrm{T}}\,\mathbf{Z}\right)_{\mathrm{H}_2+1} < \delta \tag{4.109}$$

Cele trei criterii luate în considerare de Wang pentru determinarea numărului de noduri ascunse sunt:

❶ δ – independenţa liniară

Numărul perceptronilor simpli din fiecare strat ascuns este determinat de numărul maxim de ieşiri δ – liniar independente. Valoarea δ depinde de raportul semnal – zgomot al datelor din setul de învăţare.

❷ Raportul determinanţilor (RD), definit după cum urmează:

$$\mathrm{RD}\left(\mathbf{V}^{\mathrm{T}}\,\mathbf{V}\right)_{\mathrm{H}_1} = \frac{\det\left(\mathbf{V}^{\mathrm{T}}\,\mathbf{V}\right)_{\mathrm{H}_1}}{\det\left(\mathbf{V}^{\mathrm{T}}\,\mathbf{V}\right)_{\mathrm{H}_1+1}} \tag{4.110}$$

Numărul perceptronilor simpli din fiecare strat ascuns este determinat în acest caz de prima creştere semnificativă a raportului (4.111) şi (4.112).

$$\mathrm{RD}(\mathrm{H}_1,\mathrm{H}_2)_{\mathbf{V}} = \frac{\det\left(\mathbf{V}^{\mathrm{T}}\,\mathbf{V}\right)_{\mathrm{H}_1,\mathrm{H}_2}}{\det\left(\mathbf{V}^{\mathrm{T}}\,\mathbf{V}\right)_{\mathrm{H}_1+1,\mathrm{H}_2}} \tag{4.111}$$

$$\mathrm{RD}(\mathrm{H}_1,\mathrm{H}_2)_{\mathbf{Z}} = \frac{\det\left(\mathbf{Z}^{\mathrm{T}}\,\mathbf{Z}\right)_{\mathrm{H}_1,\mathrm{H}_2}}{\det\left(\mathbf{Z}^{\mathrm{T}}\,\mathbf{Z}\right)_{\mathrm{H}_1,\mathrm{H}_2+1}} \tag{4.112}$$

Indicii determinanţilor indică dimensiunile structurii luate în considerare, de exemplu, $det\left(\boldsymbol{Z}^{\mathrm{T}}\,\boldsymbol{Z}\right)_{\mathrm{H}_1,\mathrm{H}_2+1}$ se referă la o structură având H_1 noduri în primul strat ascuns şi H_2 în cel de-al doilea.

❸ Funcţia Q, definită de relaţia (4.113).

Scanned with OKEN Scanner

$$Q(H_1) = \frac{\log\left[\det\left(\mathbf{V}^T\,\mathbf{V}\right)_{H_1}\right] - \log\left[\det\left(\mathbf{V}^T\,\mathbf{V}\right)_{H_1+1}\right]}{H_1 + 1} \qquad (4.113)$$

Numărul neuronilor din fiecare strat ascuns se poate alege prin determinarea valorii maxime a funcţiei Q, şi anume *max* $Q(H)$.

Algoritmul de construire a reţelei este constituit din paşii prezentaţi în tab. 4.13.

Tab. 4.13 Algoritmul Wang

❶ Se alege setul de date de învăţare (date de intrare şi ieşiri dorite). ❷ Se alege structura iniţială a reţelei, cu două straturi ascunse având fiecare dimensiunea H_1 şi H_2 (de exemplu, $H_1 = H_2 = 2$). ❸ Se iniţializează ponderile reţelei cu valori aleatoare; reţeaua învaţă până se obţine convergenţa. ❹ Se aplică unul dintre criteriile de performanţă de mai sus, de exemplu criteriul de δ – liniar independenţă. Se calculează $det\left(\boldsymbol{V}^T\,\boldsymbol{V}\right)_{H_1}$ şi $det\left(\boldsymbol{Z}^T\,\boldsymbol{Z}\right)_{H_2}$. ❺ Test conform organigramei din fig. 4.15. ❻ Dacă pentru ambele straturi este îndeplinit criteriul de oprire, atunci STOP.

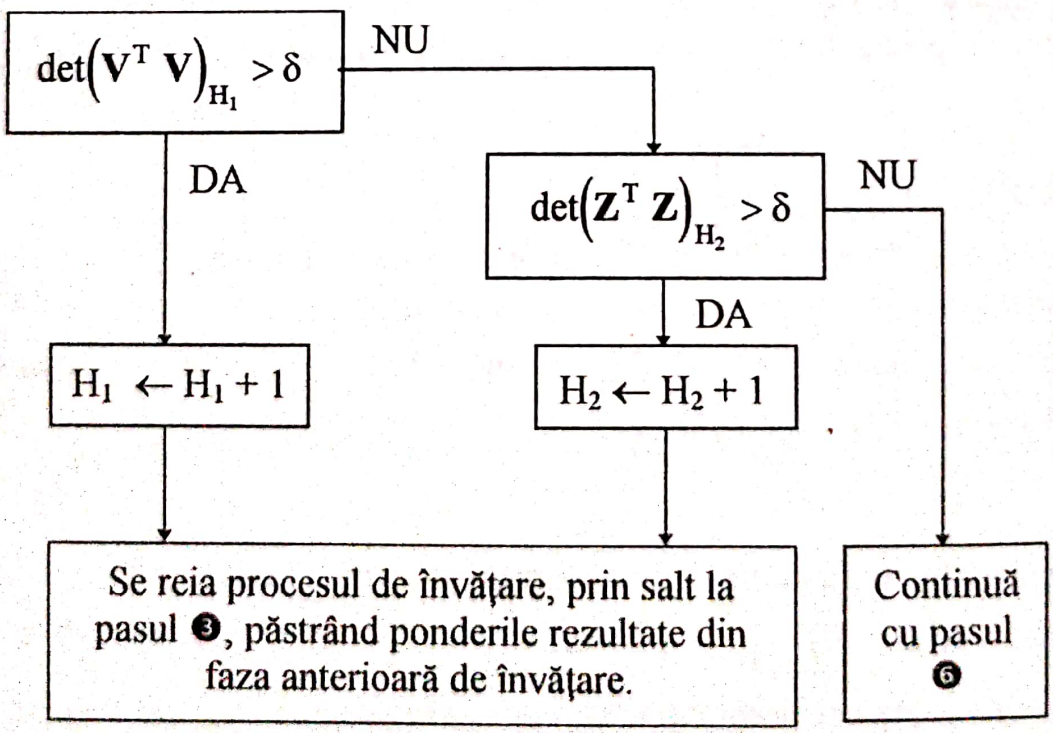


Fig. 4.15 Organigrama testelor efectuate în pasul ❺ al algoritmului

Scanned with OKEN Scanner

Algoritmul a fost aplicat pentru estimarea concentraţiei de biomasă în procesul de fermentaţie a penicilinei industriale. Din utilizarea criteriilor de dimensionare, s-a observat că funcţia Q permite identificarea valorii maxime cu uşurinţă. Cu toate că nu se observă diferenţe semnificative privind dimensiunea finală a reţelei, în funcţie de criteriul ales, Wang nu a realizat comparaţia criteriilor. Folosirea acestora în paralel permite verificarea alegerii unei structuri, în sensul validării alegerii pe baza unui criteriu, cu ajutorul celorlalte.

Metoda este simplă. Valoarea parametrului δ se stabileşte pe baza raportului semnal – zgomot din datele de învăţare. In cazul problemei studiate de Wang, se alege $\delta = 10^{-4}$, dar nu se indică modul de determinare a acestei valori.

4.5.10 Comparaţie între metodele ontogenice constructive pentru proiectarea structurilor "feedforward"

❒ ***Comparaţie* (I)**

Fiesler [F28] a propus o prezentare compactă sub formă de tabel, a unui număr de 32 de modele ontogenetice, dintre care o parte sunt constructive. Clasificarea este realizată considerând mai întâi criteriul legii de învăţare (de tip perceptron, auto – organizare, etc.), apoi după criteriul topologiei (multistrat sau arbore) şi, în sfârşit, după tipul metodei, constructivă / "distructivă" / constructivă – "distructivă". Pentru fiecare metodă se precizează tipul acesteia, apoi tipul elementului adăugat / eliminat (nod / strat / conexiune). Se marchează dacă metoda este locală, deci bazată pe informaţia existentă în perceptronul simplu respectiv, incluzând şi informaţii legate de vecinii săi. De asemenea, se indică precizarea sau nu a condiţiilor iniţiale. Tipul topologiei este marcat "F" pentru reţele total interconectate ("fully interconnected"), "T" pentru structuri de tip arbore ("tree") sau "O" ("others") pentru alte structuri. De asemenea, se precizează dacă există o demonstraţie a convergenţei, de tip "M" (matematic), "I" (informală) sau nu există.

❒ ***Comparaţie* (II)**

Frattale Mascioli, Martinelli, ş.a. [M59] au realizat o comparaţie între 4 algoritmi de creştere a reţelei, şi anume Upstart, CASCOR şi OSA. Problema de test comună este clasificarea celor două spirale.

Criteriile de comparaţie sunt: capacitatea de generalizare, costul învăţării, supraînvăţarea şi complexitatea structurii. Generalizarea a fost evaluată prin inspectarea regiunii de decizie dintre cele două spirale. Costul învăţării este direct proporţional cu timpul consumat pentru faza de învăţare. Supraînvăţarea este evitată de două dintre metode, şi anume LPPA şi OSA, care în plus, reduc

Scanned with OKEN Scanner

structura neurală. Complexitatea structurii se evaluează prin numărul total al perceptronilor simpli din structura rezultată. Rezultatele comparaţiei sunt prezentate succint în tab. 4.14.

Tab. 4.14 Rezultatele comparaţiei algoritmilor Upstart, CASCOR, LPPA, OSA

Algoritm	Generalizare (ordinea performanţei)	Costul învăţării (PC 486)	Supra învăţarea (ierarhie)	Complexitatea structurii (nr. noduri)
Upstart	2	< 1 oră	2	18
CASCOR	2	< 1 oră	2	18
LPPA	1	< 5 min.	1	21
OSA	1	< 5 s	1	45

❒ ***Comparaţie* (III)**

Propunem o clasificare succintă sub forma tab. 4.15, a metodelor constructive prezentate în cadrul prezentului capitol. Numele metodei s-a indicat şi prin prescurtarea sa, dacă aceasta este folosită curent. Semnificaţia indicatorilor este următoarea:

S / N Adaugă **S**trat / **N**euron
In Numărul straturilor iniţiale
T / A **T**otal interconectată sau **A**ltfel
R Valori iniţiale aleatoare ("**R**andom")
G Generalizare
◷ Timp
C Complexitate a structurii
B "Benchmark" (probleme de test)

Semnificaţia indicatorilor generalizare, timp şi complexitate a structurii este aceeaşi ca şi în cazul comparaţiei precedente. Intrucât problemele de test sunt diferite, aceasta este posibilă numai calitativ, marcând prin semnele 👍, 👎 şi ≈ rezultatele bune, rezultatele mai slabe, respectiv pe cele aproximativ identice sau apropiate de cele ale altor algoritmi (de obicei BKP, CASCOR).

S-a notat cu XOR problema SAU – EXCLUSIV.

Metodele constructive prezintă avantaje şi dezavantaje. Este semnificativ faptul că, realizarea unei generalizări corespunzătoare, ulterior învăţării într-un interval de timp rezonabil a unei structuri de o complexitate cât mai redusă, presupune realizarea unui compromis între indicatorii menţionaţi (tab. 4.15)

Scanned with OKEN Scanner

Tab. 4.15 Metode ontogenice constructive pentru proiectarea rețelelor neurale "feedforward"

Metoda constructivă	Autori	Anul	Nume metodă	S / N	In	T/A	R	G	🕒	C	B
Ad hoc	Ash	1989	"Dynamic Node Creation"	N		T	R	≈	≈	≈	
	Chang			N				≈	≈	≈	
	Honavar & Uhr	1988, 1989	"Generation"	S	L	A		≈	≈	≈	
	Mezard & Nadal	1989	"Tiling"	S, N	2	T	R	≈	≈	≈	
Metode de partiționare a spațiului de intrare	Reilly		RCE	N	3	A		👍	👎	👍	XOR
	Alpaydin	1990	GAL	N	3	A		👍	👎	👍	XOR
	Masciolli & Martinelli	1994	OSA	N		T	R	👍	👍	👎	Două spirale
Metode de corectare a erorii	Fahlman	1990, 1991	CASCOR	S	2	T	R	👍	👍	👍	XOR Două spirale Paritate *n*–dimensională
	Refenes	1991	CL	N				👍	👍	👍	Două spirale

Scanned with OKEN Scanner

Tab. 4.15 Metode ontogenice constructive pentru proiectarea rețelelor neurale "feedforward" (cont.)

Metoda constructivă	Autori	Anul	Nume metodă	S / N	In	T/A	R	G	🕒	C	B
Metode de corectare a erorii (cont.)	Masciolli & Martinelli	1994	LPPA	S, N	2			👍	👍	👎	Două spirale
	Ivakhnenko	1968	GMDH	S	1			👎	👎	👍	Predicție Regresie
Metode care generează arbori	Frean	1990	UPSTART	N	Arbore	A		≈	👍	👍	Două spirale
	Nahban & Zomaya	1994		N	Arbore	A		≈	👍	👍	Aproximarea funcțiilor polinomiale
Metode care pornesc de la un arbore inițial	Sethi	1995	Rețele entropice	N				≈	👎	👍	XOR
	Brent	1995		N				≈	👍	≈	Clasificarea vocalelor
Metode de modularizare a rețelei	Hanson & Pratt	1990		N				≈	👍	👎	Clasificarea vocalelor
	Lange & Voigt	1994	TACOMA	S				≈	👍	👎	Două spirale

Scanned with OKEN Scanner

Avantajele metodelor ontogenice constructive sunt:

❶ Metodele constructive pornesc de la o structură de dimensiuni reduse. Majoritatea timpului consumat pentru proiectarea arhitecturii şi estimarea valorilor ponderilor, implică o structură mică, de aceea costul acestor metode este mai redus decât al metodelor "distructive" (cap. 4.6).

❷ Costul proiectării, măsurat în numărul operaţiilor aritmetice efectuate pe un calculator serial, este $O(H)$, cu H numărul parametrilor independenţi.

❸ Toate metodele constructive obţin detectori de trăsături (noduri ascunse) cu performanţe bune pentru problema de studiu, fără a construi reţele de dimensiuni foarte mari, cu excepţia unor cazuri izolate.

❹ De obicei, utilizarea nodurilor ascunse cu funcţii de activare complexe are drept urmare introducerea fenomenului de "îngheţare" a ponderilor conexiunilor noului nod introdus în structură, în scopul limitării consumului de timp.

Dezavantajele metodelor ontogenice constructive sunt:

❶ Algoritmii care presupun interconectarea totală a structurii determină deseori fenomenul de supraînvăţare (excepţie OSA, LPPA). Soluţia este reducerea unor conexiuni, prin metode prezentate în cap. 4.6.

❷ Metodele care modifică toate ponderile au dezavantajul apariţiei fenomenului "ţintei mişcătoare" [F18, F20]. Intuitiv, fiecare nod ascuns tinde să evolueze astfel încât să devină un detector de trăsături util, dar procesul este dificil datorită faptului că toate nodurile ascunse îşi modifică ponderile în acelaşi timp. De aceea, metodele care îngheaţă ponderile nodului / stratului introdus în reţea, după adăugarea acestuia în structură, sau care permit modificarea valorii unei singure ponderi la un moment dat, obţin performanţe mai bune.

❸ Metodele care generează arbori şi cele care construiesc structura neurală pornind de la un arbore iniţial sunt consumatoare de timp, în comparaţie cu alte tipuri de metode.

❹ Metodele care implică utilizarea mai multor noduri – candidat, dintre care se alege cel mai bun sunt consumatoare de resurse (memorie), dar oferă performanţe mai bune din punct de vedere al complexităţii structurii generate.

❺ Nu este demonstrată convergenţa tuturor algoritmilor constructivi.

❻ Evaluarea exactă a performanţelor tuturor metodelor ontogenice constructive este dificilă, întrucât problemele de test sunt diferite.

❼ Nici una dintre metodele existente nu este optimală.

Scanned with OKEN Scanner

4.6 METODE ONTOGENICE "DISTRUCTIVE"

Metodele ontogenice de reducere a dimensiunilor reţelei neurale, numite şi metode descrescătoare sau "distructive" ("pruning methods"), consideră o structură iniţială mare, din care se elimină în timpul învăţării sau după terminarea acesteia, conexiuni sau noduri. Structura iniţială are dimensiuni mari, pentru a fi capabilă să înveţe rapid; structura compactă rezultată în urma reducerii, are trei avantaje:

- Complexitate mai mică, deci generalizează mai bine
- Complexitatea operaţiilor matematice se micşorează
- Structura se adaptează mai rapid în cazul modificărilor mediului de funcţionare

Deseori, după eliminarea unor conexiuni / noduri este necesară parcurgerea unei noi etape de învăţare.

Metodele descrescătoare se bazează pe algoritmi numiţi "distructivi", datorită antonimiei cu termenul "constructiv" utilizat în cazul proiectării structurilor crescătoare, şi nicidecum faptului că, alterează structura sau performanţele arhitecturii neurale.

Strategia de reducere a structurii, pentru rezolvarea problemei generale (4.1)[17] este cea numită "Lama lui Ockham" şi exprimă principiul parsimoniei (par. 4.1).

4.6.1 Formularea problemei

Problema generală de reducere a structurii poate fi formulată ca determinarea modelului de complexitate mai mică decât a celui iniţial, ale cărui performanţe se menţin într-un interval impus.

Se consideră o RPM, cu M, H şi N noduri de intrare, ascunse şi respectiv, de ieşire. Fără a restrânge generalitatea, se consideră $N = 1$. Vectorul celor Π parametri (ponderi şi praguri) ai reţelei este $\mathbf{w} = [w_1, ..., w_r, ..., w_\Pi]^T$, unde w_r este componenta indice r a vectorului, $1 \le r \le \Pi$. Funcţia implementată de reţea este F_w. Presupunem că relaţia dintre matricile ieşirii dorite $\boldsymbol{D}$ şi intrării $\boldsymbol{X}$ poate fi reprezentată prin relaţia:

$$\mathbf{D} = F(\mathbf{X}) + \mathbf{N} \qquad (4.114)$$

unde s-a notat cu $\boldsymbol{N}$ matricea valorilor zgomotului, cu medie zero şi dispersie finită (matrice având drept coloane, vectorii $\boldsymbol{n}$), iar funcţia F este o funcţie cu valori reale, definită pe $\mathfrak{R}^n$. Coloanele matricilor $\boldsymbol{X}$, $\boldsymbol{D}$ sunt patternurile de intrare şi, respectiv, de ieşire dorite. Din proprietatea de aproximare a reţelelor

[17] Minimizează capacitatea reţelei, cu constrângerile că, este capabilă să înveţe setul datelor de învăţare [G15].

Scanned with OKEN Scanner

neurale de tip "feedforward", rezultă că funcţia F poate fi aproximată prin funcţia $F_{\mathbf{w}}$, realizată de reţea şi relaţia (4.114) devine:

$$\mathbf{D} = F_{\mathbf{w}}(\mathbf{X}) + \Xi + \mathbf{N} \tag{4.115}$$

unde Ξ marchează eroarea datorată aproximării. Dacă se notează $\Xi + N = \Phi$, atunci modelul rezultat $\mathbf{M}(\mathbf{w})$ este definit de expresia (4.116), unde Φ este matricea zgomotului total, cu medie nulă şi dispersie finită, σ^2 [C25].

$$\mathbf{D} = F_{\mathbf{w}}(\mathbf{X}) + \Phi \tag{4.116}$$

Se consideră P observaţii independente $(x(\xi), d(\xi))$, $1 \le \xi \le P$, ale perechii (x, d). Estimatul $\hat{\mathbf{w}}$ al parametrilor $\mathbf{w}$, în sensul normei L_2, este definit ca vectorul parametrilor, care minimizează funcţia cost (4.117). Invăţarea reţelei constă în determinarea estimatului $\hat{\mathbf{w}}$.

$$C(\mathbf{w}) = \sum_{\xi=1}^{P} \left[d(\xi) - F_{\mathbf{w}}\left(\mathbf{X}(\xi)\right) \right]^2 \tag{4.117}$$

Ştergerea unei ponderi w_q, $1 \le q \le \Pi$, este echivalentă cu luarea în considerare a ipotezelor de test $w_q = 0$ şi $w_q \neq 0$, sau, altfel spus, testarea modelului $\mathbf{M}(\mathbf{w}_{(q)})$ faţă de $\mathbf{M}(\mathbf{w})$. Vectorii de parametri $\mathbf{w}_{(q)}$ şi $\mathbf{w}$ sunt identici, cu excepţia ponderii w_q, care este 0 în vectorul $\mathbf{w}_{(q)}$ şi nenulă în $\mathbf{w}$.

Ştergerea succesivă a ponderilor w_{q_1}, w_{q_2}, etc., este echivalentă testării modelelor $\mathbf{M}\left(\mathbf{w}_{(q_1)}\right)$, $\mathbf{M}\left(\mathbf{w}_{(q_2)}\right)$, etc., unde succesiv $w_{q_1} = 0$, apoi $w_{q_1} = w_{q_2} = 0$, etc.

Din punct de vedere al strategiei de test, problemele la care trebuie formulate răspunsuri sunt:

- Cum se realizează testul modelului $\mathbf{M}(\mathbf{w}_{(q)})$ faţă de $\mathbf{M}(\mathbf{w})$?
- Cum se determină cea mai bună secvenţă $q_1, q_2, \ldots q_L$?
- Cum se alege criteriul de oprire ?

Se urmăreşte obţinerea unei structuri de dimensiuni cât mai mici, care să generalizeze cât mai bine. Arhitectura rezultată este de obicei de acelaşi tip ca şi cea iniţială, dar modul de interconectare diferă (poate fi totală sau nu pentru structura iniţială şi este parţială pentru cea finală). Trebuie determinate însă şi următoarele aspecte:

- Este arhitectura finală simetrică sau nu ?
- Cum se alege arhitectura iniţială astfel încât arhitectura finală să fie minimă şi să generalizeze corespunzător ?

Scanned with OKEN Scanner

4.6.2 Clasificarea metodelor ontogenice "distructive"

O sinteză a metodelor "distructive" este realizată în lucrările [R14] şi [H30]. Clasificarea metodelor ontogenice de reducere a structurii neurale se poate realiza în funcţie de mai multe criterii: strategia "distructivă" abordată (tab. 4.16, 4.17), problema de studiu abordată, criteriul de oprire a reducerii structurii.

Intre tipurile de metode prezentate în tab. 4.17 există zone de suprapunere, întrucât funcţia cost poate conţine, de exemplu, termeni de senzitivitate. Diferenţa majoră între acestea se referă la faptul că, în timp ce metodele de estimare a senzitivităţii modifică o structură care deja a învăţat, variantele bazate pe adăugarea unor termeni în funcţia cost funcţionează diferit: prezenţa acestora determină eliminarea în timpul învăţării a unor ponderi care nu sunt necesare. Chiar dacă uneori nu sunt eliminate în mod real, ci doar au valori mult mai mici decât cele iniţiale, totuşi reţeaua se comportă ca un sistem de dimensiuni mai reduse.

O caracteristică a metodelor de reducere a structurii neurale este aceea că, varietatea problemelor de test ("benchmark") este foarte mare. De aceea, comparaţia diferiţilor algoritmi "distructivi" este dificilă, aplicaţiile fiind foarte diferite.

Alegerea criteriului de oprire a reducerii structurii se poate face pe baza metodelor statistice, care asigură obţinerea generalizării dorite. In acest caz, clasificarea devine cea din cap. 4.4.

In continuare vom considera clasificarea propusă în tab. 4.18.

Tab. 4.16 Clasificarea metodelor "distructive" în funcţie de strategia de reducere a structurii (varianta I)

Metode "distructive"	■ Elimină ponderi în exces
	■ Elimină noduri din reţea

Tab. 4.17 Clasificarea metodelor "distructive" în funcţie de strategia de reducere a structurii (varianta II)

Metode "distructive"	■ Bazate pe estimarea senzitivităţii funcţiei eroare, la îndepărtarea unui element (pondere, nod) din structură; elementele structurale care o afectează cel mai puţin, sunt eliminate.
	■ Bazate pe adăugarea unui termen suplimentar în funcţia cost, care să recompenseze reţeaua în cazul alegerii soluţiei corecte.

Scanned with OKEN Scanner

Tab. 4.18 Clasificarea propusă a metodelor ontogenice "distructive"

Metode ontogenice de reducere a dimensiunilor reţelei neurale	
Metode care elimină ponderi în exces	■ Metode drastice ■ Metode de grupare a ponderilor ■ Bazate pe estimarea senzitivităţii funcţiei eroare, la îndepărtarea unui element din structură ■ Bazate pe adăugarea unui termen suplimentar în funcţia cost (regularizare)
Metode care elimină noduri din reţea	■ Metode drastice ■ Metode de grupare a ponderilor ■ Bazate pe estimarea senzitivităţii funcţiei eroare, la îndepărtarea unui element din structură ■ Bazate pe adăugarea unui termen suplimentar în funcţia cost (regularizare) ■ Alte metode

4.6.3 Metode ontogenice "distructive" care elimină ponderi în exces

4.6.3.1 Metode drastice

❒ ***Metodă drastică* (I)**

Thodberg [G15] utilizează o metodă drastică de reducere a structurii, care atribuie valoarea zero, pe rând, câte unei ponderi, şi apoi evaluează modificarea ce apare în eroarea la ieşire. Dacă aceasta creşte semnificativ, atunci se reface conexiunea (se atribuie ponderii valoarea iniţială), altfel valoarea acesteia rămâne zero.

Pe un calculator serial, fiecare propagare în sens direct consumă un interval de timp de ordinul $O(\Pi)$, unde Π este numărul total de ponderi. Operaţia de anulare a unei ponderi se repetă pentru fiecare dintre acestea, şi pentru fiecare pattern din setul de învăţare. Timpul total necesar unui pas de descreştere a reţelei este de ordinul $O(M\,\Pi^2)$.

❒ ***Metodă drastică* (II)**

O variantă a metodei precedente este aceea care evaluează întâi modificarea erorii datorată fiecărei ponderi, pentru fiecare pattern din setul de învăţare. Ponderea care are efectul minim asupra acesteia, se anulează. Operaţia trebuie repetată până când nu mai există nici o modificare sesizabilă a erorii (rămâne

aproximativ constantă).

Metoda este lentă, necesitând un timp de calcul de ordinul $O(M\,H^3)$.

❒ ***Metodă drastică* (III)**

Se consideră că, un nod ascuns nu contribuie esenţial la obţinerea soluţiei, deci poate fi eliminat, dacă valoarea sa de ieşire este aproximativ constantă pentru toate patternurile de învăţare (atunci acel nod acţionează ca un prag suplimentar pentru toţi perceptronii simpli cu ale căror intrări se conectează).

❒ ***Metodă drastică* (IV)**

Dacă două noduri ascunse furnizează aceeaşi valoare de ieşire, cu acelaşi semn sau cu semn schimbat, pentru toate patternurile de învăţare, atunci numai unul dintre aceste noduri este necesar.

4.6.3.2 Metode de grupare a ponderilor

❒ ***Metodă de tip "local bottleneck"*(I)**

Kruschke [R14] a propus o metodă, numită metoda "ştrangulare locală" ("local bottleneck"), care se bazează pe competiţia nodurilor ascunse, rezultatul fiind păstrarea acestora sau eliminarea din structură. Gradul în care un perceptron simplu contribuie la realizarea funcţiei intrare – ieşire a reţelei, este măsurat de norma vectorului său de ponderi. Aceasta este considerată ca un parametru suplimentar, numit "câştig" ("gain"), iar vectorul de ponderi este normalizat. Un nod cu câştigul nul are ieşirea constantă, contribuind numai prin valoarea de prag la funcţionarea structurii.

Perceptronii simpli sunt redundanţi dacă vectorii de ponderi sunt aproape paraleli sau antiparaleli şi concurează cu alţi vectori care au direcţii similare. Câştigurile c sunt modificate conform relaţiei (4.118), unde s–au notat cu:

$\mathbf{u}_i(\xi)$ vectorul unitate, cu direcţia $\mathbf{w}_i(\xi)$, pentru patternul de intrare ξ

$\mathbf{w}_i$ vectorul ponderilor conexiunilor care intră în nodul i

$\langle\bullet\rangle$ produsul scalar

γ constantă pozitivă, cu valoare mică

$$\Delta c_i(\xi) = -\gamma \sum_{j\neq i} \cos^2\left[\angle\left\langle \mathbf{w}_i(\xi), \mathbf{w}_j(\xi)\right\rangle\right] c_j(\xi) = -\gamma \sum_{j\neq i}\left\langle \mathbf{u}_i(\xi), \mathbf{u}_j(\xi)\right\rangle^2 c_j(\xi) \tag{4.118}$$

Dacă nodul *i* are vectorul de ponderi paralel cu cel al nodului *j*, atunci câştigul fiecăruia va descreşte proporţional cu câştigul celuilalt nod. Intrucât câştigul este o valoare pozitivă, rezultă că valoarea sa minimă este zero şi atunci, perceptronul simplu poate fi eliminat din structură când câştigul său devine nul.

Competiţia nodurilor este combinată cu propagarea inversă a erorii. Prima modifică valorile câştigurilor, cea din urmă modifică valorile ponderilor. După fiecare ciclu BKP, ponderile sunt normalizate.

Metoda prezentată reduce structura reţelei prin anularea câştigurilor nodurilor. Parametrul γ este cel care stabileşte în ce măsură sunt folosite atât legea BKP, cât şi legea competitivă amintită, iar valoarea sa trebuie aleasă: valori prea mari ale parametrului determină eliminarea prea multor perceptroni simpli, prin dominarea legii competitive.

❒ *Metode de tip "distributed bottleneck"* (II)

Kruschke [K501] a sugerat o metodă, numită metoda "ştrangulare distribuită" ("distributed bottleneck"). Ideea de bază este reducerea dimensiunii spaţiului scanat de vectorii de ponderi ai perceptronilor simpli ascunşi şi formarea unor grupuri de vectori ai ponderilor în subspaţiul obţinut, a cărui dimensiune este mai mică.

Spaţiul iniţial al ponderilor reţelei cu un strat ascuns (ponderile iniţiale sunt aleatoare) poate fi redus dacă se introduc anumite constrângeri. Efectul este similar reducerii structurii, deşi numărul nodurilor şi ponderilor rămâne acelaşi, iar redundanţa şi toleranţa la defecte se păstrează.

Se notează cu d_{ij} distanţa dintre vectorii $\mathbf{w}_i$ şi $\mathbf{w}_j$ (vectori ai ponderilor conexiunilor care intră în nodurile *i*, respectiv *j*):

$$d_{ij} = \| \mathbf{w}_i - \mathbf{w}_j \| \tag{4.119}$$

Iniţial există H vectori, cu media zero şi separare (distanţă medie) iniţială D. La fiecare pas, distanţa medie calculată global este dată de relaţia (4.120). După fiecare pas de propagare inversă a erorii, ponderile sunt modificate conform relaţiei (4.121).

$$\bar{d} = \frac{2}{H(H-1)} \sum_{i<j} d_{ij} \tag{4.120}$$

$$\mathbf{w}_i(k+1) = \mathbf{w}_i(k) + \beta \sum_{j \neq i} \left(d_{ij} - \bar{d}\right)\left(\mathbf{w}_i - \mathbf{w}_j\right) \tag{4.121}$$

Dacă $d_{ij} > \bar{d}$, atunci vectorul $\mathbf{w}_i$ este deplasat mai departe de $\mathbf{w}_j$.

Scanned with OKEN Scanner

Dacă $d_{ij} < \bar{d}$, atunci vectorul $\mathbf{w}_i$ este deplasat mai aproape de $\mathbf{w}_j$.

Vectorii sunt apoi normalizaţi din nou, astfel încât media devine din nou zero, iar separarea egală cu D, distanţa medie iniţială.

Algoritmul este echivalent aplicării metodei gradientului negativ asupra funcţiei eroare, pe o suprafaţă dată, cu constrângeri.

❒ *Metode de tip "soft weight sharing"* (III)

Metoda propusă de Nowlan şi Hinton [H30] sub numele de "soft weight sharing", se bazează pe ipoteza că, ponderile au o distribuţie apriori mixtă gaussiană, iar funcţia cost ce trebuie minimizată conţine doi termeni:

$$C = \frac{P}{2\sigma_y^2}\sum_{\xi=1}^{P}\left[d(\xi) - y(\xi)\right]^2 - \sum_{r=1}^{\Pi}\log\left(\sum_{b=1}^{B}\tau_b\, p_b(w_r)\right) \qquad (4.122)$$

unde σ_y^2 este dispersia erorii pătratice. Reţeaua conţine un singur nod de ieşire ($N = 1$), iar vectorul parametrilor săi este $\mathbf{w} = [w_1, ..., w_r, ..., w_\Pi]^T$, cu $1 \le r \le \Pi$. Fiecare termen $p_b(w_r)$ este o funcţie densitate de probabilitate gaussiană, cu media μ_b şi dispersia σ_b^2, iar τ_b este proporţia în care a intrat în amestec funcţia densitate de probabilitate indice b, cu $1 \le b \le B$.

Prin derivarea funcţiei cost C în raport cu ponderea w_r, se obţine:

$$\frac{\partial C}{\partial w_r} = \frac{P}{2\sigma_y^2}\sum_{\xi}\left[d(\xi) - y(\xi)\right]^2 \frac{\partial y(\xi)}{\partial w_r} - \sum_{r} a_b(w_r)\frac{\mu_b - w_r}{\sigma_b^2} \qquad (4.123)$$

S–a notat cu $a_b(w_r)$ probabilitatea aposteriori dată de expresia (4.124), când se cunoaşte ponderea w_r. Similar, se poate calcula gradientul în raport cu ceilalţi parametri şi rezultă relaţia (4.125).

$$a_b(w_r) = \frac{\tau_b\, p_b(w_r)}{\sum_q \tau_q\, p_q(w_r)} \qquad (4.124)$$

$$\frac{\partial C}{\partial \mu_b} = \sum_b a_b(w_r)\frac{\mu_b - w_r}{\sigma_b^2} \qquad (4.125)$$

Condiţia pentru aflarea valorii optime a mediei μ_b, notată cu μ_b^*, este $\partial C/\partial \mu_b = 0$ şi rezultă relaţia (4.126).

$$\mu_b^* = \frac{\sum_r a_b(w_r)\, w_r}{\sum_r a_b(w_r)} \tag{4.126}$$

Gradientul în raport cu σ_b este dat de relaţia (4.127), iar condiţia $\partial C/\partial \sigma_b = 0$ are soluţia (4.128). Pentru parametrul τ_b se poate scrie relaţia (4.129), iar estimatul optimal este dat de (4.130).

$$\frac{\partial C}{\partial \sigma_b} = -\sum_r a_b(w_r) \frac{(\mu_b - w_r)^2 - \sigma_b^2}{\sigma_b^3} \tag{4.127}$$

$$(\sigma_b^*)^2 = \frac{\sum_r a_b(w_r)(\mu_b^* - w_r)^2}{\sum_r a_b(w_r)} \tag{4.128}$$

$$\frac{\partial C}{\partial \tau_b} = \sum_r \left(1 - \frac{a_b(w_r)}{\tau_b}\right) \tag{4.129}$$

$$\tau_b^* = \frac{\sum_r a_b(w_r)}{\sum_{ni} 1} \tag{4.130}$$

4.6.3.3 Metode bazate pe estimarea senzitivităţii

Metodele de acest tip se bazează pe estimarea influenţei pe care o are eliminarea unei conexiuni asupra funcţiei cost a structurii. Numită generic senzitivitate, aceasta influenţă este definită şi denumită diferit (salienţă — "saliency", senzitivitate — "sensitivity", evidenţă — "evidence", relevanţă — "relevance", etc.) în metodele prezentate în continuare.

❒ ***Metoda OBD ("Optimal Brain Damage") (I)***

Denker, Le Cun, Sara Solla, ş.a. [D200] au sugerat o metodă de reducere a numărului ponderilor pentru o RPM care a învăţat, metodă numită OBD ("Optimal Brain Damage"). Ideea de bază este estimarea creşterii erorii de

Scanned with OKEN Scanner

învăţare, în cazul anulării unor ponderi. Estimatul acesteia este exprimat în funcţie de senzitivitatea (sau caracteristica — salienţa) funcţiei cost (eroare) la eliminarea ponderii respective, care este un produs dintre aproximarea derivatei de ordinul doi a funcţiei eroare în raport cu ponderile şi pătratul valorii ponderii. Ponderile care determină modificări minime ale erorii, deci au salienţă redusă, sunt anulate definitiv. Trebuie menţionat faptul că, sintagma "salienţa ponderii", utilizată atât de referinţele bibliografice, cât şi în continuare, reprezintă exprimarea mai simplă, dar inexactă, a expresiei "salienţa funcţiei cost la eliminarea ponderii".

Paşii menţionaţi se realizează după ce reţeaua a învăţat, de aceea o primă ipoteză este că, funcţia cost are o valoare de minim local, gradientul său fiind nul în punctul respectiv. Salienţa ponderii reprezintă cu aproximaţie, curbura funcţiei eroare în subspaţiul scanat de fiecare vector pondere (fig. 4.16).

Pentru reţeaua cu un singur nod de ieşire ($N = 1$), având vectorul parametrilor $\mathbf{w} = [w_1, ..., w_r, ..., w_\Pi]^T$, valorile ieşirii efective $y(\xi)$ şi respectiv, ieşirii dorite $d(\xi)$, pentru toate patternurile ξ, modificarea introdusă în eroarea de învăţare de perturbarea vectorului de ponderi $\mathbf{w}$, prin eliminarea unei componente, este:

$$\delta C_{\hat{inv}}(\mathbf{w}) = C_{\hat{inv}}(\mathbf{w} + \delta\mathbf{w}) - C_{\hat{inv}}(\mathbf{w}) =$$

$$= \left(\nabla C_{\hat{inv}}\right)(\delta \mathbf{w}) + \sum_{r=1}^{\Pi}\sum_{q=1}^{\Pi} \frac{\partial^2 C_{\hat{inv}}}{\partial w_r \, \partial w_q} dw_r \, dw_q + O\left(\|\delta \mathbf{w}\|^3\right) =$$

$$= \sum_{r=1}^{\Pi} \frac{\partial C_{\hat{inv}}}{\partial w_r} dw_r + \frac{1}{2}\sum_{r=1}^{\Pi} \frac{\partial^2 C_{\hat{inv}}}{\partial w_r^2} dw_r^2 + \frac{1}{2}\sum_{r\neq q}^{\Pi} \frac{\partial^2 C_{\hat{inv}}}{\partial w_r \, \partial w_q} dw_r \, dw_q +$$

$$+ O\left(\|\delta \mathbf{w}\|^3\right) \qquad (4.131)$$

S-a presupus funcţia cost (eroare de învăţare) dată de relaţia (3.62) (normă L_2). Derivata de ordinul doi a funcţiei cost în raport cu ponderile este elementul (r, q) al matricii hessiene $\boldsymbol{H}$.

Cele trei ipoteze de bază ale metodei sunt:

❶ Funcţia cost (eroare) de învăţare se află într-un punct de minim local (reţeaua a învăţat), deci $\nabla C_{\hat{inv}} = 0$.

❷ Termenii de ordin mai mare sau egal cu trei din dezvoltarea (4.131) se pot neglija.

Scanned with OKEN Scanner

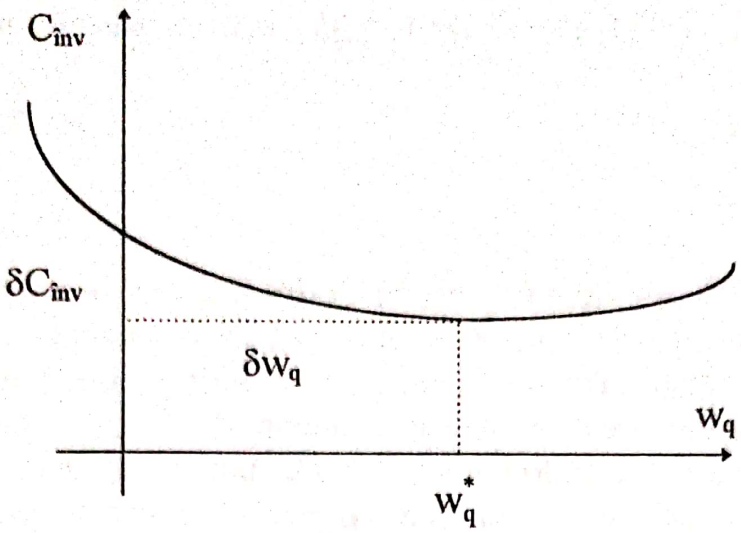


Fig. 4.16 După învăţare, se presupune că eroarea de învăţare a reţelei a atins un punct de minim local, cu $w_q = w_q^*$. Această pondere este eliminată prin atribuirea valorii $w_q = 0$. Modificarea funcţiei cost datorită acestei operaţii este $\delta C_{înv}$, estimată de valoarea salienţei

❸ Matricea hessiană are dimensiuni mari şi termenii aflaţi în afara diagonalei principale $\left(\partial^2 C_{înv} / \partial w_r \, \partial w_q\right)$ se pot neglija. Un alt motiv este acela că, se presupune eliminarea câte unei singure ponderi, deci $\delta w_r = 0$ pentru toţi r, cu excepţia $\delta w_q = -w_q$, adică $r = q$.
Pe baza ipotezelor ❶ – ❸, relaţia (4.131) devine:

$$\delta C_{înv} \approx \frac{1}{2} \sum_q \frac{\partial^2 C_{înv}}{\partial w_q^2} \left(d w_q\right)^2 \qquad (4.132)$$

Folosind aproximarea Levenberg – Marquardt pentru matricea hessiană, se sugerează că derivata de ordinul doi a funcţiei cost $C_{înv}$ poate fi aproximată. Atunci, creşterea erorii de învăţare poate fi exprimată ca (4.133) sau, în funcţie de salienţa fiecărei ponderi, conform relaţiei (4.134), cu indicele q aparţinând setului $\mathcal{D}$, ce marchează ponderile $\{w_q \mid q \in \mathcal{D}\}$, care urmează să fie şterse.

$$\delta C_{înv} \approx \sum_q \frac{1}{2} \frac{\partial^2 C_{înv}}{\partial w_q^2} w_q^2 \qquad (4.133)$$

$$\delta C_{înv} = \sum_{q \in \mathcal{D}} s_q \qquad (4.134)$$

Scanned with OKEN Scanner

Din relaţiile (4.133) şi (4.134) rezultă valoarea salienţei pentru ponderea w_q:

$$s_q \cong \frac{1}{2} \frac{\partial^2 C_{\hat{i}nv}}{\partial w_q^2} w_q^2 \qquad (4.135)$$

Metoda OBD este iterativă. Paşii algoritmului sunt prezentaţi în tab. 4.19. Se poate observa că, în pasul ❸ se elimină câte o pondere. Numărul total al conexiunilor eliminate depinde de condiţia de oprire impusă. Aceasta se poate baza pe alegerea empirică a unui număr maxim de ponderi care se pot elimina (de exemplu, 5% din numărul total). O altă posibilitate este oprirea anulării ponderilor atunci când reţeaua nu mai reuşeşte să implementeze corect setul de date de învăţare. De asemenea, criteriile statistice (cap. 4.4) pot fi utilizate pentru stabilirea unui criteriu de oprire.

Tab. 4.19 Paşii algoritmului OBD

Repetă ❶ Reţeaua învaţă, până când $C_{\hat{i}nv} < \varepsilon$, cu ε dat. ❷ Se calculează valorile salienţei fiecărei ponderi. Se ordonează ponderile în ordinea descrescătoare a salienţelor. ❸ Se elimină conexiunea care are ponderea cu salienţa cea mai scăzută. ❹ Salt la pasul ❶. □ Până când condiţia de STOP este îndeplinită.

Relaţiile generale (4.132) – (4.135) se pot detalia, pentru cazurile funcţiei eroare aditivă şi apoi, pentru cazul particular al unei structuri RPM cu un singur strat ascuns (două straturi active, cel ascuns şi stratul de ieşire).

Dacă funcţia cost este aditivă, atunci eroarea de învăţare se poate scrie ca (4.136). Elementele diagonale ale matricii hessiene a funcţiei cost sunt date de relaţia (4.137), unde $y(\xi) = F_w(x(\xi))$.

$$C_{\hat{i}nv} = \frac{1}{P} \sum_{\xi=1}^{P} C_{\hat{i}nv}(\xi) \qquad (4.136)$$

Pentru funcţia cost totală (4.136), în care $C_{\hat{i}nv}(\xi)$ este dată de relaţia (3.62), expresia (4.137) devine (4.138).

Scanned with OKEN Scanner

$$\frac{\partial^2 C_{\hat{i}nv}}{\partial w_q^2} = \frac{1}{P}\sum_{\xi=1}^{P}\left[\frac{\partial^2 C_{\hat{i}nv}(\xi)}{\partial\left(y(\xi)\right)^2}\left(\frac{\partial F_w\left(x(\xi)\right)}{\partial w_q}\right)^2 + \frac{\partial C_{\hat{i}nv}(\xi)}{\partial y(\xi)}\,\frac{\partial^2 F_w\left(x(\xi)\right)}{\partial w_q^2}\right] \tag{4.137}$$

$$\frac{\partial^2 C_{\hat{i}nv}}{\partial w_q^2} = \frac{1}{P}\sum_{\xi=1}^{P}\left\{\left(\frac{\partial F_w\left(x(\xi)\right)}{\partial w_q}\right)^2 + \left[d(\xi) - F_w\left(x(\xi)\right)\right]\left(-\frac{\partial^2 F_w\left(x(\xi)\right)}{\partial w_q^2}\right)\right\} \tag{4.138}$$

Aproximarea Levenberg – Marquardt invocată de Denker, Le Cun ş.a. [D200] abandonează cel de-al doilea termen. Dacă reţeaua a învăţat corect exemplele din setul de învăţare, atunci eroarea $e(\xi) = d(\xi) - F_w(x(\xi))$ este zero şi se justifică acest lucru. Totuşi, principalul motiv pentru renunţarea la al doilea termen este legat de simplificarea calculului.

Aproximaţia finală a creşterii erorii de învăţare datorată eliminării ponderii w_q este (4.139). Dacă $C_{\hat{i}nv}$ este suma pătratelor erorilor, atunci $\partial^2 C_{\hat{i}nv}/\partial y^2$ este constantă.

$$s_q = w_q^2\,\frac{1}{2P}\sum_{\xi=1}^{P}\frac{\partial^2 C_{\hat{i}nv}(\xi)}{\partial\left(y(\xi)\right)^2}\left(\frac{\partial F_w\left(x(\xi)\right)}{\partial w_q}\right)^2 \tag{4.139}$$

Pentru cazul în care RPM are două straturi active (un singur strat ascuns), cu intrare M – dimensională, şi H noduri ascunse şi $N = 1$ (un singur nod de ieşire), se pot rescrie relaţiile (4.136) – (4.139) pentru a se obţine expresiile explicite ale salienţelor ponderilor w_{ih} dintre stratul de intrare şi cel ascuns, respectiv ale ponderilor $v_h \overset{def.}{=} v_{hj}\big|_{j=1}$ dintre stratul ascuns şi cel de ieşire, unde $s_h \overset{def.}{=} s_{hj}\big|_{j=1}$. Pragurile există sub forma unor ponderi suplimentare unitare. Se presupune funcţia cost considerată anterior.

Funcţia globală implementată de reţea este $F_{w,v}$, dată de (4.140), notată prescurtat F_w iar derivata sa în raport cu ponderile dintre stratul ascuns şi cel de ieşire este (4.141). Salienţa unei astfel de ponderi este (4.142):

$$F_w\left(x(\xi)\right) = \sum_{h=1}^{H} v_h \tanh\left(\sum_{i=0}^{M} w_{ih}\, x_i(\xi)\right) + v_0 \tag{4.140}$$

$$\frac{\partial F_w(\mathbf{x}(\xi))}{\partial v_h} = \tanh\left(\sum_{i=0}^{M} w_{ih}\, x_i(\xi)\right) \quad , \text{cu } 1 \le h \le H \tag{4.141}$$

$$s_h = v_h^2 \; \frac{1}{2P} \sum_{\xi=1}^{P} \tanh^2\left(\sum_{i=0}^{M} w_{ih}\, x_i(\xi)\right), \text{cu } 1 \le h \le H \tag{4.142}$$

Derivata funcţiei F_w în raport cu ponderile dintre stratul de intrare şi cel ascuns este (4.143), cu $1 \le i \le M$ şi $1 \le h \le H$, iar salienţa unei astfel de ponderi este (4.144).

Valorile salienţelor reflectă distribuţia ponderilor reţelei şi senzitivitatea unei anumite funcţii realizate de reţea la distribuţia patternurilor de intrare.

$$\frac{\partial F_w(\mathbf{x}(\xi))}{\partial w_{ih}} = v_h \left[1 - \tanh^2\left(\sum_{m=0}^{M} w_{mh}\, x_m(\xi)\right)\right] x_i(\xi) \tag{4.143}$$

$$s_{ih} = v_h^2\, w_{ih}^2 \; \frac{1}{2P} \sum_{\xi=1}^{P} \left[1 - \tanh^2\left(\sum_{m=1}^{M} w_{mh}\, x_m(\xi)\right)\right]^2 \tag{4.144}$$

Metoda OBD a fost aplicată de LeCun, Denker şi Solla pentru reducerea unor reţele care fuseseră deja optimizate manual. Gorodkin, ş.a., au realizat un studiu cantitativ al metodei, pentru problema de clasificare "contiguity problem" (numită şi problema "two or three clumbs"[18]) şi pentru "problema călugărilor" ("monks problem[19]"). Funcţia cost considerată a fost regularizată, parametrul de regularizare fiind $\varepsilon = 0.0005$ (4.145), iar modificarea ponderilor dintre stratul de intrare şi cel ascuns se face folosind relaţia (4.146):

$$C = C_{inv} + C_r = \frac{1}{2P} \sum_{\xi=1}^{P} \left[d(\xi) - F_w(\mathbf{x}(\xi))\right]^2 + \frac{\varepsilon}{2P} \sum_{q_1=1}^{\Pi_N} \left(v_{q_1}\right)^2 \tag{4.145}$$

$$\Delta w_{ih} = -\eta \left(\frac{\partial C_{inv}}{\partial w_{ih}} + \frac{\varepsilon}{P} w_{ih}\right) \Bigg/ \left(\frac{\partial^2 C_{inv}}{\partial w_{ih}^2} + \frac{\varepsilon}{P}\right) \tag{4.146}$$

---

[18] Câmpul de intrare este interpretat ca o "imagine" unidimensională, în care se numără componentele de valoare +1. Se consideră două tipuri de patternuri: cele care au două şi cele care au trei componente. Funcţia booleană are valoarea de ieşire −1 dacă intrarea are 2 componente, altfel +1. Se studiază cazul cu 10 intrări (792 valori de intrare posibile, dintre care 432 au 3 componente şi 360 au 2 componente) [G15].

[19] Problema numită monk1 a fost formulată pentru roboţi artificiali, descrişi de 6 atribute diferite. Este tot o problemă de clasificare booleană [G15].

Scanned with OKEN Scanner

S-a notat cu Π_N numărul ponderilor conexiunilor dintre stratul ascuns şi cel de ieşire.

Metoda a fost aplicată şi pentru predicţia seriilor de timp Mackey – Glass [H27], precum şi a seriei de timp "pete solare". S-a considerat, de asemenea, o funcţie cost regularizată.

Rezultatele arată că metoda OBD funcţionează. Curba de învăţare se îmbunătăţeşte, structura rezultată este caracterizată de matrici de ponderi rare, iar arhitectura neurală finală este rareori simetrică. Generalizarea este mai bună pentru structura finală, după ce aceasta a parcurs încă o etapă de învăţare.

Imbunătăţirea performanţelor se poate realiza prin determinarea adaptivă a parametrilor de regularizare, sau prin aplicarea metodei OBD asupra unui ansamblu de reţele neurale, urmată de selectarea reţelei de dimensiuni minime. Aplicarea unui simplu vot majoritar asupra ansamblului este de asemenea, suficientă pentru obţinerea unor curbe de învăţare mai bune, unui număr mai mic de patternuri de învăţare necesare (150 în ultimul caz şi 100 anterior, faţă de 200 pentru reţeaua iniţială).

Trebuie remarcat faptul că, înaintea aplicării metodei pentru rezolvarea unei probleme date, trebuie precizat dacă aceasta poate fi sau nu, rezolvată de o reţea care nu este total interconectată, sau altfel spus, pentru care reţeaua total interconectată este redundantă. De exemplu, orice problemă de clasificare booleană, este rezolvabilă cu o reţea a cărei matrice de ponderi este rară.

Printre avantajele metodei se numără:

❶ Imbunătăţirea curbei de învăţare prin aplicarea metodei OBD.

❷ Prin inspectarea structurilor obţinute se observă că, reţeaua cu mai puţini parametri are eroarea de test cea mai mică şi astfel, se confirmă ideea "Lama lui Ockham".

❸ Aproximaţiile pe care se bazează metoda OBD sunt valide pentru problemele de clasificare luate în considerare.

❹ In timp ce reţelele total interconectate se pot utiliza numai dacă există un set de învăţare suficient, metoda OBD se poate aplica dacă există seturi de învăţare de diferite dimensiuni.

Dezavantajele metodei sunt:

❶ Aproximaţia iniţială a erorii de învăţare, prin dezvoltarea de ordinul doi în jurul punctului de minim, chiar şi la distanţă de acesta, nu este valabilă în multe situaţii. Dacă, de exemplu, se utilizează funcţia de activare tangentă hiperbolică pentru nodurile ascunse, şi dacă aceasta se saturează pentru majoritatea intrărilor, atunci suprafaţa de eroare va fi plată, iar aproximaţia de ordinul doi departe de situaţia reală. Se poate utiliza o regularizare cu un parametru de valoare mică ("weight decay"), pentru a evita saturarea sigmoidei. Performanţele metodei se îmbunătăţesc semnificativ în acest caz.

❷ Cea de-a doua aproximaţie pe care se bazează metoda OBD se referă la faptul că, o creştere mare a erorii după eliminarea ponderii w_q şi înainte de a

Scanned with OKEN Scanner

parcurge o nouă etapă de învăţare ($\delta C_{\text{înainte}}(w_q)$), înseamnă totodată o creştere mare a erorii după reînvăţare ($\delta C_{\text{după}}(w_q)$). Acest lucru poate să nu fie adevărat. In mod ideal, ar trebui utilizată cea de-a doua valoare. Trebuie remarcat însă faptul că, $\delta C_{\text{înainte}}$ este o limită superioară pentru $\delta C_{\text{după}}$, întrucât eroarea totdeauna descreşte în timpul reînvăţării:

$$\delta C_{\text{după}}(w_q) \leq \delta C_{\text{înainte}}(w_q) \tag{4.147}$$

❸ Aproximarea diagonală a matricii hessiene este adevărată dacă se elimină câte o singură pondere, întrucât în acest caz, termenii din afara diagonalei principale nu intervin în calcul. In situaţiile în care se doreşte eliminarea mai multor ponderi odată, este necesară luarea în considerare a întregii matrici şi aplicarea unor metode de calcul exact, care sunt însă consumatoare de timp şi de resurse.

❹ Hassibi şi Stork [H501] au criticat reducerea dimensiunilor reţelei bazată pe valorile modulelor ("magnitude based pruning"), în care ponderile sunt ordonate în funcţie de valoarea salienţei. Pe baza relaţiilor (4.142) sau (4.144), se poate susţine acest punct de vedere, întrucât această ordonare este echivalentă presupunerii că derivata de ordinul doi a funcţiei cost este constantă pentru toate ponderile. Pentru ponderile dintre stratul ascuns şi ieşire, derivata de ordinul doi este determinată de nivelele de activare ale nodurilor ascunse, care nu trebuie să fie aceleaşi pentru toate nodurile.

Se remarcă de asemenea că, o valoare mică a salienţei nu presupune neapărat o valoare redusă a ponderii.

❒ *Metoda OCD ("Optimal Cell Damage")* (II)

Cibias, Soulié, Gallinari, ş.a. [C26] au propus o variantă a metodei OBD, numită OCD ("Optimal Cell Damage"). Ideea de bază este cea de selecţie a variabilelor de intrare, problemă dificilă în recunoaşterea formelor. Metoda OCD evaluează cele mai utile variabile de intrare, renunţând la cele mai puţin utile. Selecţia variabilelor este considerată ca o extensie a metodei de reducere a ponderilor.

Renunţarea la variabila de intrare x_i presupune anularea tuturor valorilor ponderilor din mulţimea Fan_Out(i) a nodului i. Costul acestei operaţii este suma costurilor eliminării fiecărei ponderi în parte, iar salienţa variabilei indice i este definită ca:

$$\psi_i = \sum_{q \in \text{Fan_Out}(q)} s_q \tag{4.148}$$

Scanned with OKEN Scanner

Dacă variaţia relativă a costului în două iteraţii succesive este prea mică, atunci se calculează valorile ψ_i, se ordonează descrescător şi se elimină b variabile de intrare, având cele mai scăzute valori ale salienţei. Pragul de decizie pentru testarea costului, precum şi valoarea b se stabilesc cu ajutorul setului de test.

Problemele abordate de Cibias, ş.a., sunt: predicţia unei serii de timp artificiale şi clasificarea formelor de undă.

Deşi perfomanţele structurii finale sunt mai bune decât cele ale structurii iniţiale, în predicţia seriilor de timp şi clasificare, nu s-a realizat comparaţia cu metoda OBD, deci nu se poate decide superioritatea algoritmului SSM.

Metoda are dezavantajul utilizării matricii hessiene în varianta aproximării sale diagonale. De asemenea, o serie de parametri trebuie stabiliţi pe baza setului de test.

❒ *Metoda OBS ("Optimal Brain Surgeon")* (III)

Metoda propusă de Hassibi şi Stork [H501], se bazează pe estimarea valorilor salienţei ponderilor, prin luarea în considerare a termenilor de ordin doi din afara diagonalei principale a matricii hessiene. Metoda OBS este o generalizare a metodei OBD.

Salienţa unei ponderi în metoda OBS este definită ca modificarea erorii de învăţare, datorată eliminării acelei ponderi, precum şi modificării prin învăţare a ponderilor rămase, în scopul obţinerii unei valori minime a funcţiei cost $C(\mathbf{w})$.

Se notează punctul de extrem al funcţiei cost, fără constrângeri, cu $\mathbf{w}_0$. Dezvoltarea funcţiei cost în jurul acestuia este:

$$C(\mathbf{w}) = C(\mathbf{w}_0) + \frac{1}{2}\,\delta\mathbf{w}^T\,\mathbf{H}\,\delta\mathbf{w} \qquad (4.149)$$

unde $\mathbf{w} = \mathbf{w}_0 + \delta\mathbf{w}$. Primul termen dispare, datorită presupunerii iniţiale că $\mathbf{w}_0$ este punct de extrem; matricea $\boldsymbol{H}$ este matricea hessiană (matricea derivatelor de ordinul doi ale funcţiei eroare de învăţare).

Următorul pas este obţinerea punctului de extrem cu constrângeri, folosind metoda multiplicatorilor Lagrange. Eliminarea ponderii indice q, cu $1 \le q \le \Pi$, se poate scrie ca (4.150), cu $\boldsymbol{u}_q$ vectorul unitate indice q.

$$\delta\mathbf{w}^T\,\mathbf{u}_q = -\,\mathbf{w}_0^T\,\mathbf{u}_q \qquad (4.150)$$

Estimatul funcţiei cost cu constrângeri se poate scrie sub forma (4.151), iar punctul de extrem este dat de relaţia (4.152). Salienţa ponderii indice q este (4.153).

Scanned with OKEN Scanner

$$\hat{C}_q(\mathbf{w}) = C(\mathbf{w}) + \gamma_q \left(\delta \mathbf{w} + \mathbf{w}_0\right)^T \mathbf{u}_q \tag{4.151}$$

$$\gamma_q = \frac{\mathbf{w}_0^T \mathbf{u}_q}{\mathbf{u}_q^T \mathbf{H}^{-1} \mathbf{u}_q} \tag{4.152}$$

$$\delta C_q(\mathbf{w}) = \frac{1}{2} \gamma_q^2 \mathbf{u}_q^T \mathbf{H}^{-1} \mathbf{u}_q \tag{4.153}$$

Estimarea salienţei prin metoda OBS se realizează conform tab. 4.20. Diferenţa faţă de metoda OBD este aceea că, în timp ce aceasta utilizează aproximarea diagonală a matricii hessiene, OBS ia în considerare întreaga matrice ***H***. Performanţele metodei în clasificare booleană sunt mai bune decât cele ale metodei OBD. Costul său este însă ridicat, datorită necesităţii calculului suplimentar al termenilor de ordin doi în matricea hessiană.

Tab. 4.20 Estimarea salienţei în metoda OBS

■ Dezvoltă funcţia cost în termeni de ordinul doi, în jurul unei valori extreme. ■ Determină modificările ponderilor rămase, ca rezultat al eliminării ponderii indice q, notată cu w_q; modifică ponderile rămase, folosind aproximări de ordinul doi. ■ Calculează modificarea în eroarea de învăţare.

❒ *Metoda SSM ("Statistical Stepwise")* (IV)

Cottrel, Girard, ş.a. [C25] au propus o altă abordare pentru eliminarea ponderilor, pornind de la observaţia că, estimatorii în sens L_2 sunt asimptotic normali, pentru o clasă generală de modele **M**, şi anume:

$$\sqrt{P}\left(\hat{\mathbf{w}} - \mathbf{w}\right) \xrightarrow{P \to \infty} N_{\mathbf{w}}\left(0, \sigma^2 \Sigma^{-1}\right) \tag{4.154}$$

Dispersia reziduală σ^2 este estimată de relaţia (4.155), Σ este matricea de informaţie, estimată $\left(\hat{\Sigma}\right)$ de (4.156) şi aproximată de (4.157).

$$\hat{\sigma}^2 = \frac{C(\hat{\mathbf{w}})}{P} \tag{4.155}$$

Scanned with OKEN Scanner

$$\hat{\Sigma} = \frac{1}{2P} \nabla^2 C(\hat{w}) \tag{4.156}$$

$$\hat{\Sigma} \approx \frac{1}{P} \sum_{\xi} \left[\nabla F_{\hat{w}}(X(\xi)) \right] \left[\nabla F_{\hat{w}}(X(\xi)) \right]^T \tag{4.157}$$

Procedura propusă de Cottrell, Girard, ş.a., este prezentată în tab. 4.21. Procesul de reducere a structurii se încheie atunci când nu mai poate fi eliminată nici o pondere, deci nu este necesar un criteriu de oprire suplimentar. S-a notat cu $\hat{w}_{(q)}$ vectorul parametrilor care conţine componenta $\hat{w}_q = 0$.

Tab. 4.21 Algoritmul SSM

❶ Se alege o structură cu vectorul Π – dimensional de parametri **w** şi se învaţă reţeaua;

❷ Se calculează toate câturile (4.160), unde deviaţia standard estimată a ponderii $\hat{w}_q$ este (4.161):

$$Q(q) = \frac{\hat{w}_q}{\hat{\sigma}(\hat{w}_q)} \tag{4.158}$$

$$\hat{\sigma}(\hat{w}_{(q)}) = \frac{\hat{\sigma}}{\sqrt{P}} \sqrt{(\hat{\Sigma}^{-1})_{q,q}} \tag{4.159}$$

❸ Se defineşte $q_1 = \min_q Q(q)$ (4.160)

❹ Se testează modelul $\mathbf{M}(\mathbf{w}_{(q_1)})$ (ipoteza nulă) faţă de $\mathbf{M}(\mathbf{w})$.

Dacă $Q(q_1) < 1.96$ (stabilit pentru o diferenţă de 5%) atunci acceptă eliminarea ponderii w_{q_1}.

❺ In cazul rejecţiei, opreşte procesul de eliminare şi păstrează modelul $\mathbf{M}$ anterior. In cazul acceptării, reinstruieşte reţeaua corespunzând modelului $\mathbf{M}(\mathbf{w}_{(q_1)})$ şi repetă pasul ❷, pornind de la $\mathbf{M}(\mathbf{w}_{(q_2)})$, căutând indicele q_2, etc.

Se arată că, metoda SSM este în relaţie de echivalenţă directă cu metoda OBD, întrucât pentru valoarea salienţei, definită de OBD ca relaţia (4.161), se poate verifica relaţia (4.162).

Scanned with OKEN Scanner

$$s_q = \frac{1}{2P}\left[C\left(\hat{w}_{(q)}\right) - C(w)\right] \tag{4.161}$$

$$Q(q)^2 = \left(\frac{\hat{w}_q}{\hat{\sigma}\left(\hat{w}_q\right)}\right)^2 = \frac{2P s_q}{C(\hat{w})/(P-\Pi)} \tag{4.162}$$

Intrucât numitorul poate fi considerat constant pentru P având valori mari, rezultă că $Q(q)^2$ şi s_q sunt proporţionale. De aceea, comparaţia conform valorilor $Q(q)$ este echivalentă comparaţiei după valorile salienţei.

Diferenţele dintre cele două metode sunt acelea că, OBD nu ia în considerare valorile relative ale salienţei şi consideră aproximarea diagonală a matricii hessiene, în timp ce metoda SSM calculează matricea hessiană completă. Astfel, este posibilă ştergerea simultană a mai multor ponderi şi evaluarea exactă a efectului, faţă de aproximarea diagonală, care este corectă numai dacă estimatorii sunt independenţi.

Principalul avantaj al metodei SSM este acela că, stabileşte în mod obiectiv ce ponderi pot fi eliminate, având valoare mică în raport cu deviaţia standard estimată. Printre dezavantaje se numără dificultatea alegerii structurii iniţiale şi dependenţa modelului final de modelul iniţial cu Π parametri. Se poate alege criteriul Akaike pentru compararea modelelor iniţial şi final.

❒ ***Metodă de calcul a senzitivităţii*** **(V)**

Karnin [K500] a propus evaluarea senzitivităţii funcţiei eroare la tăierea unei conexiuni. Conexiunile pentru care care senzitivitatea funcţiei eroare este cea mai scăzută sunt eliminate (ponderile sunt anulate). Senzitivitatea unei ponderi w_q, cu $1 \le q \le \Pi$, este definită astfel:

$$s_q = -\frac{C\left(w^{\hat{inv}}\right) - C\left(w^{final}\right)}{w^{\hat{inv}} - w^{final}} w^{\hat{inv}} = -\frac{C\left(w^{\hat{inv}}\right) - C(0)}{w^{\hat{inv}} - 0} w^{\hat{inv}} \tag{4.163}$$

unde s–au notat cu:

s_q senzitivitatea unei ponderi w_q
$w^{\hat{inv}}$ ponderea conexiunii după etapa de învăţare
w^{final} ponderea conexiunii după tăierea acesteia (valoare zero)
$C(w^{\hat{inv}})$ eroarea după faza de învăţare, înainte de tăierea conexiunii
$C(w^{final})$ eroarea după tăierea conexiunii (= $C(0)$)

Scanned with OKEN Scanner

Se propune evaluarea senzitivităţii s prin sumarea tuturor modificărilor ponderii în timpul învăţării (în locul calculului direct al valorii $C(0)$ după tăierea unei conexiuni). Valoarea estimată a senzitivităţii este:

$$\hat{s}_q = -\sum_{k=0}^{K-1} \frac{\partial C}{\partial w_q} \Delta w_q(k) \frac{w_q^{inv}}{w_q^{inv} - w_q^{initial}} \tag{4.164}$$

unde s–a notat cu K numărul total de epoci de învăţare, iar $w_q^{initial}$ este valoarea iniţială (înaintea învăţării) a ponderii indice q (componentă a vectorului Π – dimensional $\mathbf{w} = [w_1, ..., w_q, ..., w_\Pi]^T$). In cazul în care modificarea ponderilor se realizează cu algoritmul BKP, rezultă:

$$\hat{s}_q = \sum_{k=0}^{K-1} \left[\Delta w_q(k) \right]^2 \frac{w_q^{inv}}{\eta \left(w_q^{inv} - w_q^{initial} \right)} \tag{4.165}$$

Ponderile având cele mai scăzute valori ale senzitivităţilor sunt eliminate.

Există cazuri particulare ale metodei: dacă toate conexiunile ce ies dintr-un nod sunt tăiate, rezultă că nodul însuşi poate fi eliminat; dacă toate ponderile conexiunilor ce intră într–un nod sunt zero, rezultă că ieşirea are valoare constantă, deci nodul poate fi eliminat, modificând totodată pragul celorlalţi perceptroni simpli.

4.6.3.4 Metode "distructive" bazate pe regularizare

Am arătat (par. 3.2.4) că, regularizarea constă în esenţă, în adăugarea unui termen suplimentar în funcţia cost, care penalizează complexitatea modelului. In interpretare bayesiană, acest termen reprezintă ipoteza asupra densităţii de probabilitate apriori a ponderilor RPM şi poate fi gaussian, Cauchy, laplacian, etc. Există o legătură directă între regularizare şi metodele de reducere a dimensiunilor reţelei, în sensul că, modificarea structurii apare uneori ca o consecinţă imediată a utilizării termenilor de penalizare. O trăsătură comună a celor două tipuri de metode este necesitatea stabilirii parametrilor de control: parametrul de regularizare pentru metodele de penalizare, respectiv criteriul de oprire pentru metodele ontogenice "distructive".

Regularizarea adaptivă propusă de Hansen, ş.a. [H26], elimină necesitatea stabilirii empirice a parametrilor de regularizare. Se arată, de asemenea, că reducerea reţelei este o consecinţă firească în structurile neurale cu regularizare adaptivă: dacă valorile parametrilor de descreştere a ponderilor tind la infinit,

Scanned with OKEN Scanner

atunci se obţine de fapt, reducerea structurii, observaţie care unifică ambele clase de metode, sub ideea generală de control al capacităţii reţelei.

O parte dintre metodele de descreştere a structurii prin regularizare, prezentate în continuare, se numesc metode cu scăderea ponderilor ("weight decay") şi se bazează pe ideea scăderii ponderilor cu valori mici, astfel încât acestea să tindă, în final, spre zero. După terminarea etapei de învăţare, conexiunile respective pot fi eliminate, obţinându-se astfel, o structură cu mai puţine grade de libertate (mai puţini parametri).

Deşi denumirea de "metode de regularizare cu scăderea ponderilor" se aplică în multe situaţii care presupun penalizarea funcţiei cost, aceasta este motivată din punct de vedere matematic numai pentru metodele în care matricea de regularizare este $\boldsymbol{R} = \chi \boldsymbol{I}$, cu $\boldsymbol{I}$ matricea unitate [D205]. De aceea, în continuare vom utiliza denumirea generică de metode "distructive" bazate pe regularizare, indicând dacă acestea sunt (sau nu) de tip "weight decay".

❒ ***Metodă "distructivă" bazată pe regularizare de tip "weight decay" şi distribuţie apriori Laplace a ponderilor*** **(I)**

Ishikawa a propus utilizarea unei funcţii de cost, în care termenul de penalizare presupune o distribuţie apriori Laplace a ponderilor:

$$C = \sum_{\xi \in \Delta} \left[d(\xi) - y(\xi) \right]^2 + \chi \sum_{q=1}^{\Pi} |w_q| \qquad (4.166)$$

unde $\Delta = \{x(\xi), d(\xi)\}$ este setul de date de învăţare. Cel de–al doilea termen al funcţiei cost adaugă în legea de modificare a ponderilor, factorul $-\chi\, sgn(w_q)$, unde *sgn* este funcţia semm, iar $1 \le q \le \Pi$. Se observă că, pentru $w_q > 0$, ponderea este decrementată cu χ, iar pentru $w_q < 0$, este incrementată cu χ.

Avantajele utilizării termenului de penalizare laplacian din punct de vedere al reducerii dimensiunilor structurii, sunt argumentate în lucrarea [W19] şi prezentate în [D205].

❒ ***Metodă "distructivă" bazată pe regularizare de tip "weight decay" şi distribuţie apriori Gauss a ponderilor*** **(II)**

Hinton [H41] a propus utilizarea unui termen de penalizare gaussian în funcţia cost globală:

$$C = C_{inv} + \chi \sum_{q=1}^{\Pi} w_q^2 \qquad (4.167)$$

Scanned with OKEN Scanner

Aplicarea metodei gradientului negativ (4.168) pentru funcţia cost (4.167), cu η rata de învăţare, este echivalentă utilizării relaţiei (4.169) după fiecare actualizare a ponderii w_q, unde ε este un parametru de valoare mică (4.170).

$$\Delta w_q = -\eta \frac{\partial C}{\partial w_q} \tag{4.168}$$

$$(w_q)_{nou} = (1 - \varepsilon)\,(w_q)_{anterior} \tag{4.169}$$

$$\varepsilon = \frac{\chi \eta}{2} \tag{4.170}$$

Relaţia (4.167) penalizează utilizarea mai multor ponderi w_q decât este necesar, descurajând totodată utilizarea ponderilor de valori mari.

❒ ***Metodă "distructivă" bazată pe regularizare de tip "weight decay"* (III)**

Din relaţia (4.167) rezultă că, o pondere de valoare mare are un cost mult mai ridicat decât mai multe ponderi de valoare mică. Pentru a rezolva această problemă, se poate utiliza un termen de penalizare diferit, dat de relaţia (4.171) [K7], situaţie echivalentă dependenţei parametrului ε de componenta w_q, $1 \le q \le \Pi$, a vectorului de ponderi **w** (relaţia (4.172)).

$$C = C_{inv} + \chi \sum_{q=1}^{\Pi} \frac{w_q^2}{1 + w_q^2} \tag{4.171}$$

$$\varepsilon_q = \frac{\chi \eta / 2}{\left(1 + w_q^2\right)^2} \tag{4.172}$$

Ponderile w_q cu valoare mică tind mai rapid către zero decât ponderile cu valori mari. Pentru cazul erorii de învăţare de tip L_2 şi normării funcţiei cost globale la numărul patternurilor de învăţare P, expresia (4.171) devine (4.173), unde primul termen este eroarea de învăţare C_{inv}, iar cel de-al doilea este un estimat al complexităţii modelului stohastic constituit de RPM.

$$C = \frac{1}{P} \sum_{\xi=1}^{P} \left[d(\xi) - y(\xi)\right]^2 + \frac{\chi}{P} \sum_{q=1}^{\Pi} \frac{w_q^2}{1 + w_q^2} \tag{4.173}$$

Scanned with OKEN Scanner

Se poate observa că, pentru ponderi $w_q >> 1$,

$$\frac{\chi}{P}\sum_{q=1}^{\Pi}\frac{w_q^2}{1+w_q^2}\approx\frac{\chi\,\Pi}{P} \tag{4.174}$$

unde Π este numărul total de parametri independenţi.

Calculând derivatele parţiale (4.175), ale celui de-al doilea termen al funcţiei cost în raport cu w_q, se observă că, termenul de regularizare penalizează ponderile cu valori mai mici, în mai mare măsură decât ponderile cu valori mari. Când valoarea ponderii w_q este suficient de mare, aceasta nu va fi modificată datorită celui de-al doilea termen al funcţiei cost:

$$\frac{\partial}{\partial w_q}\left(\sum_{q=1}^{\Pi}\frac{w_q^2}{1+w_q^2}\right)=\frac{d}{d\,w_q}\left(\frac{w_q^2}{1+w_q^2}\right)=\frac{2\,w_q}{\left(1+w_q^2\right)^2}\to\begin{cases}0 & \text{când } w_q\to\infty\\ \cong 2\,w_q & \text{când } w_q\to 0\end{cases} \tag{4.175}$$

Intrucât iniţial, toate ponderile au valori mici, este important ca parametrul χ să aibă de asemenea, o valoarea mică, pentru a permite modificarea valorilor acestora. Apoi, valoarea χ poate să crească pentru a se elimina ponderile cu valori mici.

Metoda prezentată poate fi considerată şi ca un caz particular al grupării ponderilor, din paragrafele anterioare. In acest caz, este favorizată o partiţie care conţine ponderile de valoare nulă, datorită prezenţei termenului de regularizare în funcţia cost.

❒ *Metodă "distructivă" bazată pe regularizare* (IV)

O variantă a metodei (III) este cea propusă de Weigend, ş.a. [H41], care minimizează funcţia cost:

$$C=\sum_{\xi\in\Delta}\left[d(\xi)-y(\xi)\right]^2+\chi\sum_{q\in\mathbf{C}}\frac{w_q^2/w_c^2}{1+w_q^2/w_c^2} \tag{4.176}$$

cu Δ setul de patternuri de învăţare şi $\mathbf{C}$ setul tuturor conexiunilor din reţea (în cazul în care pragurile sunt considerate ponderi constante ale unor conexiuni suplimentare introduse în structura neurală, relaţiile $q \in \mathbf{C}$ şi $1 \le q \le \Pi$ sunt echivalente, unde s-a notat cu Π numărul parametrilor reţelei). Cel de–al doilea termen exprimă complexitatea reţelei în funcţie de valorile ponderilor, în raport

cu valoarea constantă w_c. Pentru $|w_q| >> w_c$, costul unei ponderi tinde la χ. Pentru $|w_q| << w_c$, costul tinde la zero.

Pentru valori mari ale parametrului χ, metoda este similară celor din categoria "weight decay". Stabilirea acestuia nu este însă simplă, fiind dependentă de aplicaţie: dacă este o valoare prea mică, atunci efectul termenului de regularizare în funcţia cost este neglijabil; dacă este o valoare prea mare, atunci toate ponderile sunt forţate să aibă valori ce tind spre zero. Weigend a propus metode empirice de modificare a parametrului de regularizare.

❒ ***Metoda OBD cu regularizare de tip "weight decay"*** **(V)**

Funcţia cost considerată de Hansen, ş.a. [H27], este:

$$C = C_{inv} + \frac{\chi_w}{P} \sum_{q_1=1}^{\Pi_H} w_{q_1}^2 + \frac{\chi_v}{P} \sum_{q_2=1}^{\Pi_N} v_{q_2}^2 \qquad (4.177)$$

unde s-au notat cu:

- χ_w parametrul de descreştere a ponderilor conexiunilor dintre stratul de intrare şi cel ascuns
- χ_v parametrul de descreştere a ponderilor conexiunilor dintre stratul ascuns şi cel de ieşire
- Π_H numărul parametrilor (ponderi, praguri) dintre stratul de intrare şi cel ascuns
- Π_N numărul parametrilor (ponderi, praguri) dintre stratul de ascuns şi cel de ieşire
- w_{q_1} ponderile conexiunilor dintre stratul de intrare şi cel ascuns, $1 \le q_1 \le \Pi_H$
- v_{q_2} ponderile conexiunilor dintre stratul ascuns şi cel de ieşire, $1 \le q_2 \le \Pi_N$

Termenul C_{inv} al funcţiei cost este de tip L_2, normat la numărul patternurilor de învăţare P.

Aplicarea metodei OBD, cu funcţia cost (4.177) conduce la relaţia (4.178) pentru creşterea erorii datorită eliminării unei ponderi şi (4.179) pentru aproximarea derivatei de ordinul doi a termenului C_{inv} în raport cu ponderile.

$$\delta C_{inv} = \sum_{q \in \mathcal{D}} s_q \approx \sum_{q \in \mathcal{D}} \left(\frac{2}{P} + \frac{1}{2} \frac{\partial^2 C_{inv}}{\partial w_q^2} \right) w_q^2 \qquad (4.178)$$

Scanned with OKEN Scanner

$$\frac{\partial^2 C_{inv}}{\partial w_q^2} \approx \frac{2}{P} \sum_{\xi=1}^{P} \left(\frac{\partial F_w(\mathbf{x}(\xi))}{\partial w_q} \right)^2 \qquad (4.179)$$

unde w_q este componenta indice q a ponderii $\mathbf{w}$, cu $1 \le q \le \Pi$.

Metoda OBD cu regularizare se aplică similar algoritmului OBD standard. In plus, parametrii de regularizare χ_w şi χ_v sunt determinaţi precis, prin regularizare adaptivă. La început, pentru reţeaua total interconectată, valorile acestora sunt ridicate, indicând preponderenţa termenului de regularizare în funcţia cost, apoi descresc, pe măsură ce numărul parametrilor structurii scade.

❒ *Metoda OBS cu regularizare de tip "weight decay"* (VI)

Hansen şi Pedersen au propus [H29] modificarea algoritmului OBS ("Optimal Brain Surgeon") prin regularizare. Funcţia cost regularizată este (3.54), în care termenul $C_{inv}(\mathbf{w})$ este eroarea pătratică medie pe întregul set de învăţare şi $\boldsymbol{R}$ matrice de regularizare. Pentru cazul simplu de scădere a ponderilor ("weight decay"), matricea $\boldsymbol{R}$ este proporţională cu matricea unitate: $\boldsymbol{R}_{ii'} = \chi \boldsymbol{I}_{ii'}$.

Valoarea salienţei unei ponderi în metoda OBS este definită ca modificarea erorii de învăţare, datorată eliminării acelei ponderi şi modificării celor rămase prin învăţare, pentru obţinerea valorii minime a funcţiei date de (3.54). Paşii de estimare ai valorilor salienţei, prezentaţi în cadrul metodei OBS se reiau pentru funcţia cost regularizată.

Se notează punctul de extrem al funcţiei cost, fără constrângeri, cu $\mathbf{w}_0$. Dezvoltarea funcţiei cost în jurul acesteia este dată de relaţia (4.180), unde $\mathbf{w} = \mathbf{w}_0 + \delta\mathbf{w}$.

$$C(\mathbf{w}) = C(\mathbf{w}_0) + \frac{1}{2} \delta\mathbf{w}^T (\mathbf{H} + \mathbf{R}) \delta\mathbf{w} \qquad (4.180)$$

Primul termen dispare, prin presupunerea iniţială că $\mathbf{w}_0$ este punct de extrem; matricea $\boldsymbol{H}$ este matricea derivatelor de ordinul doi ale funcţiei eroare de învăţare, iar matricea hessiană a funcţiei cost regularizată este $\boldsymbol{A} = \boldsymbol{H} + \boldsymbol{R}$.

Obţinerea extremului cu constrângeri se realizează prin aplicarea metodei multiplicatorilor Lagrange [H29]. Eliminarea ponderii indice q, w_q, se poate scrie ca relaţia (4.150), iar costul eliminării acestei ponderi, ca (4.151). Punctul de extrem este dat (4.152). Salienţa ponderii indice w_q se calculează cu relaţia:

$$\delta C_q(\mathbf{w}) = \gamma_q \, \mathbf{w}_0^T \mathbf{R} \mathbf{A}^{-1} \mathbf{u}_q + \frac{1}{2} \gamma_q^2 \, \mathbf{u}_q^T \mathbf{A}^{-1} \mathbf{H} \mathbf{A}^{-1} \mathbf{u}_q \qquad (4.181)$$

Scanned with OKEN Scanner

Expresia (4.181) se reduce la (4.153), obţinută de Hassibi şi Stork, pentru $\boldsymbol{R} = 0$ (în lipsa regularizării).

❒ ***Metodă "distructivă" bazată pe regularizare* (VII)**

Chauvin [C501] a propus utilizarea funcţiei cost:

$$C = \lambda \sum_{\xi=1}^{P} \sum_{j=1}^{N} \left[d_j(\xi) - y_j(\xi) \right]^2 + \chi \sum_{\xi=1}^{P} \sum_{h=1}^{H} \varphi\left(z_h^2(\xi) \right) \qquad (4.182)$$

unde φ este o funcţie monotonă cu valori pozitive, iar λ şi χ sunt doi parametri de ponderare a termenilor din componenţa funcţiei cost. S-a notat cu $z_h(\xi)$ ieşirea oricărui nod ascuns şi cu $y_j(\xi)$ ieşirea oricărui nod de ieşire, ambele pentru patternul de intrare ξ.

Primul termen este eroarea de învăţare de tip norma L_2, iar cel de-al doilea evaluează în ce măsură valoarea ieşirii nodurilor ascunse variază în timpul învăţării. Dacă aceasta se modifică mult, atunci nodul codifică informaţie semnificativă, altfel importanţa sa nu este mare.

Rezultatul utilizării funcţiei cost (4.182) depinde de alegerea funcţiei φ. Aceasta poate avea, de exemplu, din familia funcţiilor cu prima derivată (4.183), unde $n \in \mathbb{Z}$.

$$\varphi' = \frac{1}{\left(1 + z^2\right)^n} \qquad (4.183)$$

Valorile particulare $n = 0, 1, 2$ au efect diferit. Pentru $n = 0$, funcţia φ este liniară; toate nodurile, indiferent de variaţia valorilor de ieşire, sunt penalizate în mod egal. Pentru $n = 1$, funcţia φ este logaritmică şi nodurile cu variaţie scăzută a valorii ieşirii sunt penalizate mai puternic decât cele cu variaţie mare a acesteia. In cazul $n = 2$, penalizarea tinde către o asimptotă, pe măsură ce variaţia valorii de ieşire a nodului creşte. Astfel, nodurile cu variaţie mare a valorii ieşirii nu sunt penalizate mult mai mult decât cele cu variaţie medie a acesteia.

Chauvin a studiat şi varianta introducerii unui termen suplimentar în funcţia cost, dependent de pătratul valorii ponderii:

$$C = \lambda \sum_{\xi=1}^{P} \sum_{j=1}^{N} \left[d_j(\xi) - y_j(\xi) \right]^2 + \chi \sum_{\xi=1}^{P} \sum_{h=1}^{H} \varphi\left(z_h^2(\xi) \right) + \mu \sum_{q=1}^{\Pi} w_q^2 \qquad (4.184)$$

Scanned with OKEN Scanner

Derivata celui de–al treilea termen în raport cu w_q este $2\,\mu w_q$, deci prezenţa acestuia în funcţia cost are efectul introducerii unei scăderi a ponderilor ("weight decay").

Modificarea funcţiei cost (4.184) în sensul introducerii unor termeni de tipul celor propuşi de Hinton [R14], a condus la expresia (4.185). Utilizarea acesteia şi a parametrilor cu valorile $\lambda = 0.1$, $\chi = 0.1$ şi $\mu = 0.001$, a determinat reducerea substanţială a numărului de perceptroni simpli ascunşi, indiferent de dimensiunea iniţială a reţelei.

$$C = \lambda \sum_{\xi=1}^{P} \sum_{j=1}^{N} \left[d_j(\xi) - y_j(\xi) \right]^2 + \chi \sum_{\xi=1}^{P} \sum_{h=1}^{H} \frac{z_h^2(\xi)}{1 + z_h^2(\xi)} + \mu \sum_{q=1}^{\Pi} \frac{w_q^2}{1 + w_q^2} \quad (4.185)$$

❒ *Metodă "distructivă" bazată pe regularizare* (VIII)

Ji, ş.a. [J500], au sugerat o modificare a funcţiei eroare pentru minimizarea numărului de noduri ascunse şi a valorilor ponderilor. Structura RPM considerată are un singur strat ascuns, un nod de intrare şi un nod liniar de ieşire. Iniţial reţeaua are mai multe noduri ascunse decât este necesar. Se calculează valoarea ieşirii:

$$y(\mathbf{w}, \mathbf{b}, x) = \sum_{h=1}^{H} v_h \; f(w_h \, x - b_h) \quad (4.186)$$

unde s-au notat cu w_h ponderile conexiunilor ce intră în nodul ascuns *h*, cu v_h ponderile conexiunilor ce ies din nodul ascuns *h*. Funcţia de activare *f* este sigmoidală.

Importanţa ("significance") unui perceptron simplu ascuns este (4.187) iar funcţia φ este (4.188). Pentru argument **w**, se obţin funcţiile φ menţionate în metodele anterioare.

$$S_h = \varphi(w_h)\, \varphi(v_h) \quad (4.187)$$

$$\varphi(u) = \frac{u^2}{1 + u^2} \quad (4.188)$$

Funcţia cost propusă de Ji, ş.a., este dată de relaţia (4.189), unde C_{inv} este de tip norma L_2 a erorii, iar C_r măsoară importanţa nodurilor:

$$C(\mathbf{w}, \mathbf{b}) = \lambda \; C_{inv}(\mathbf{w}, \mathbf{b}) + \chi \, C_r(\mathbf{w}) =$$

Scanned with OKEN Scanner

$$= \lambda \sum_{\xi=1}^{P} \left[y(\mathbf{w}, \mathbf{b}, x(\xi)) - d(\xi) \right]^2 + \chi \sum_{h=1}^{H} \sum_{j=1}^{h-1} S_h S_j \qquad (4.189)$$

Notaţiile sunt cele anterioare, iar λ şi χ sunt termeni de ponderare a termenilor funcţiei cost.

Modificări posibile ale metodei includ:

■ Termenul de regularizare poate fi introdus în funcţia cost, după ce reţeaua a învăţat destul de bine setul de date de învăţare. Astfel, se elimină posibilitatea unui conflict între termenii $C_{\hat{i}nv}$ şi C_r, care să conducă la apariţia unui minim local.

■ Parametrul χ poate varia, ca funcţie de $C_{\hat{i}nv}$ (4.190). Astfel, pentru valori mari ale termenului de cost $C_{\hat{i}nv}$, χ are o valoare mică, şi reciproc.

$$\chi = \chi_0 \exp\left(-\beta C_{\hat{i}nv}\right) \qquad (4.190)$$

■ Scrierea legii de modificare a ponderilor şi pragurilor conform relaţiilor (4.191), (4.192) este o variantă care favorizează ponderile cu valori mici:

$$w_q(k+1) = w_q(k) - \lambda \frac{\partial C_{\hat{i}nv}}{\partial w_q}(w(k), b(k)) - \chi \frac{\partial C_r}{\partial w_q}(w(k)) - \psi \tanh\left(w_q(k)\right) \qquad (4.191)$$

$$b_q(k+1) = b_q(k) - \lambda \frac{\partial C_{\hat{i}nv}}{\partial w_q}(w(k), b(k)) - \psi \tanh\left(b_q(k)\right) \qquad (4.192)$$

Parametrul ψ descreşte treptat, devenind nul atunci când valoarea termenului $C_{\hat{i}nv}$ nu se mai modifică (4.193). Când se atinge un nivel acceptabil al erorii, se pot elimina ponderile cu valori mici şi apoi se reia învăţarea.

$$\psi = \psi_0 \left| C_{\hat{i}nv}(w(k), b(k)) - C_{\hat{i}nv}(w(k-1), b(k-1)) \right| \qquad (4.193)$$

❒ *Metodă "distructivă" bazată pe regularizare* (IX)

Nowlan şi Hinton [H39] au propus un termen de penalizare mai complex, care modelează funcţia densitate de probabilitate a ponderilor, ca un amestec de gaussiene. Metoda a fost numită "soft weight sharing" şi a fost prezentată ca metodă de grupare a ponderilor. Seturile de ponderi puţin probabile au un cost ridicat. In consecinţă, ponderile tind să se conformeze distribuţiei în timpul

Scanned with OKEN Scanner

învăţării. Dacă distribuţia ponderilor este compusă, de exemplu, din două gaussiene, cu dispersii diferite (o distribuţie "îngustă" şi o distribuţie "largă"), ambele cu medie nulă şi amestecate în proporţii egale, atunci gaussiana îngustă are un efect puternic asupra ponderilor cu valoare mică, atrăgându-le spre valoarea zero. Influenţa sa asupra ponderilor cu valori mari este redusă, acestea fiind influenţate de către distribuţia gaussiană "largă".

- ***Metodă "distructivă" bazată pe regularizare* (X)**

MacKay [H26] a reluat abordarea Bayes a regularizării adaptive în contextul reţelelor neurale, demonstrând că metodele bazate pe calcularea unor valori de evidenţă ("evidence") pot îmbunătăţi generalizarea. El a propus o schemă Bayes de regularizare compusă din doi paşi. Aceasta a fost aplicată de Hansen, ş.a., pentru cazul estimării mediei unei variabile aleatoare, având dispersia σ^2 a distribuţiei cunoscută. S-a presupus că distribuţia apriori a valorilor medii dorite este normală. Deşi expresia evidenţei este interesantă din punct de vedere statistic, nu este clară relaţia dintre maximizarea acesteia şi minimizarea erorii de generalizare [H26].

- ***Metoda CSDF ("Convergence Suppression and Divergence Facilitation")* (XI)**

Metoda CSDF ("Convergence Suppression and Divergence Facilitation") propusă de Yasui, Malinowski şi Zurada [Y8] şi prezentată în par. 4.6.4.2, aparţine şi clasei metodelor de reducere prin regularizare, întrucât relaţia de modificare a ponderilor presupune de fapt, adăugarea în expresia funcţiei cost, a termenului de penalizare dat de relaţia (4.194), cu ε, γ constante pozitive; h indică un nod din stratul ascuns, iar j un nod de ieşire.

$$C_r = -\varepsilon \sum_{\substack{l=1 \\ l \neq h}}^{H} \frac{w_{lj}^2}{2} - \gamma\, C_{\hat{i}nv} \sum_{\substack{l=1 \\ l \neq j}}^{N} \frac{w_{hl}^2}{2} \qquad (4.194)$$

4.6.4 Metode "distructive" de eliminare a nodurilor ascunse în exces

4.6.4.1 Metode drastice

O metodă drastică de micşorare a dimensiunilor reţelei neurale "feedforward" este eliminarea directă a nodurilor ascunse, în cazul îndeplinirii unor criterii de redundanţă în nodurile de ieşire.

❒ ***Metodă drastică* (I)**

Siestma şi Dow [H30] au fost printre primii care au demonstrat funcţionarea unei metode manuale, interactive, de reducere a structurii neurale, bazată pe redundanţa existentă între nodurile de ieşire.

Reţeaua multistrat este compusă din perceptroni simpli liniari. Proiectantul inspectează reţeaua care a învăţat şi decide care noduri trebuie eliminate. Se propun o serie de metode empirice pentru identificarea nodurilor care nu participă efectiv la soluţia corectă.

Dacă un nod are ieşirea constantă pentru toate patternurile de învăţare prezentate la intrarea reţelei, atunci acesta nu contribuie la soluţia problemei şi poate fi eliminat. Totuşi, poate contribui la stabilirea valorilor de prag ale neuronilor din straturile următoare, astfel încât acestea trebuie modificate.

Dacă un număr de noduri au valorile de ieşire puternic corelate (de exemplu, identice, sau de valori opuse) pentru toate patternurile de învăţare, atunci sunt redundante şi pot fi înlocuite printr-un singur nod. Toate ponderile conexiunilor ce ies din acesta trebuie adunate, astfel încât efectul asupra nodurilor din straturile următoare să fie acelaşi.

Intrucât este puţin probabil ca valorile de ieşire ale neuronilor să fie constante dacă nodurile sunt sigmoidale, deci corelarea să fie exactă, rezultă că aplicarea regulilor empirice trebuie făcută cu atenţie.

O a doua etapă de reducere a structurii este cea în care se elimină nodurile care, deşi sunt liniar independente de nodurile din acelaşi strat, nu sunt neapărat necesare.

Exemplul oferit de Siestma şi Dow se referă la 4 patternuri de învăţare şi un strat cu trei noduri binare. De vreme ce două noduri sunt suficiente pentru codarea a patru patternuri, unul dintre cei trei neuroni poate fi eliminat. Este însă posibil, pe de altă parte, să se introducă inseparabilitate liniară prin această metodă, pentru că, de exemplu, stratul următor ar putea să fie nevoit să realizeze funcţia XOR pentru două noduri. De aceea, există şi soluţia de rezervă ca, în astfel de situaţii, să poată fi adăugat un nou strat ascuns. In ansamblu, reţeaua tinde să se transforme într–o structură mai îngustă, în comparaţie cu aspectul iniţial.

4.6.4.2 Metode de grupare a ponderilor

❒ ***Metodă de skeletonizare* (I)**

Metoda de skeletonizare propusă de Mozer [M502] poate fi considerată atât o metodă de grupare a ponderilor, cât şi o metodă bazată pe calculul senzitivităţii. Am prezentat-o în cea de-a doua clasă de metode (par. 4.6.4.3).

Scanned with OKEN Scanner

❒ *Metoda CSDF ("Convergence Suppression and Divergence Facilitation")* (II)

Yasui, Malinowski şi Zurada [Y8] au propus metoda CSDF ("Convergence Suppression and Divergence Facilitation"), de reducere a ponderilor, a cărei idee de bază este forţarea utilizării fiecărui nod ascuns de către de două sau mai multe noduri de ieşire. Reducerea ponderilor conexiunilor dintre stratul ascuns şi cel de ieşire se realizează în prezenţa a două acţiuni opuse: "suprimarea convergenţei" şi "facilitarea divergenţei", care determină şi denumirea metodei. Acţiunea de facilitare a divergenţei se înregistrează asupra intrărilor fiecărui nod de ieşire (fig. 4.17, culoare deschisă), iar acţiunea de suprimare a convergenţei asupra ieşirilor fiecărui nod ascuns (fig. 4.17, culoare închisă).

Legea de modificare a ponderilor pe baza algoritmului standard BKP devine:

$$\Delta w_{hj}(k) = -\eta \frac{\partial C_{\hat{i}nv}}{\partial w_{hj}} + \alpha \ \Delta w_{hj}(k-1) + Sup + Fac \qquad (4.195)$$

unde $\Delta w_{hj}(k)$ este variaţia ponderii w_{hj} (a conexiunii dintre nodurile h şi j), la pasul k. Primii doi termeni constituie legea de modificare standard pe baza algoritmului BKP cu moment. Ultimii doi termeni sunt definiţi astfel:

$$Sup = -\varepsilon \, sgn(w_{hj}) \sum_{\substack{l=1 \\ l \neq h}}^{H} |w_{lj}| \qquad (4.196)$$

$$Fac = -\gamma \frac{\partial C_{\hat{i}nv}}{\partial w_{hj}} \sum_{\substack{l=1 \\ l \neq j}}^{N} |w_{hl}| \qquad (4.197)$$

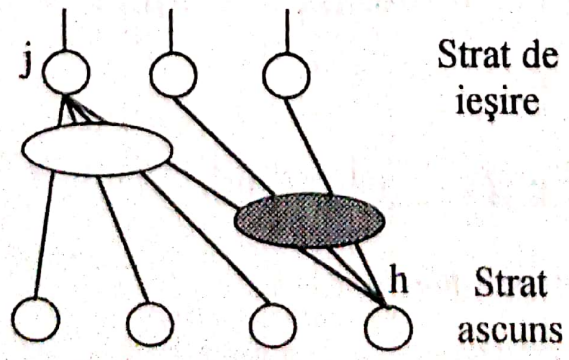


Fig. 4.17 Câmpul de suprimare a convergenţei şi cel de facilitare a divergenţei

Valorile η, α, ε şi γ sunt constante, iar C_{inv} este eroarea de învăţare în sens L_2. Termenul *Sup* realizează "suprimarea", în sensul împiedicării creşterii valorilor ponderilor conexiunilor dintre stratul ascuns şi cel ieşire. Acestea acţionează reciproc, una asupra celeilalte, pentru realizarea acestui efect, iar rezultatul este de fapt o inhibiţie laterală parametrică. Ideea de inspiraţie provine din domeniul biologic.

Termenul *Fac* realizează "facilitarea" convergenţei, în sensul că din fiecare nod ascuns pornesc conexiuni care încearcă să contribuie reciproc la creşterea valorilor ponderilor, accentuând efectul algoritmului BKP.

Dacă acţiunea celor două efecte opuse este prea puternică, atunci legea BKP nu poate acţiona efectiv şi conexiunea respectivă dispare.

Metoda este aplicată pentru clasificarea datelor IRIS, precum şi pentru o clasificare a imaginilor bitmap. Prin comparaţie cu structura fixă care învaţă cu algoritmul BKP standard, metoda permite reducerea reţelei în timpul învăţării, numărul ponderilor dintre stratul de intrare şi cel ascuns scade cu aproximativ 95%, iar cel al ponderilor dintre stratul ascuns şi cel de ieşire cu aproximativ 90%.

Un avantaj al metodei este faptul că, prin reducerea structurii în timpul învăţării, rezultă o arhitectură neurală de dimensiuni mici comparativ cu cele iniţiale. Performanţele nu se modifică în sens negativ.

Dezavantajele metodei se referă la necesitatea stabilirii experimentale a valorilor parametrilor η, ε, γ, α. De asemenea, se constată senzitivitatea rezultatelor la modificarea parametrilor ε şi γ.

4.6.4.3 Metode bazate pe calculul senzitivităţii

Metodele de acest tip se bazează pe estimarea influenţei (numită generic, senzitivitate) pe care o are eliminarea unui perceptron simplu asupra funcţiei cost a structurii.

❒ ***Metodă de skeletonizare* (I)**

Mozer şi Smolensky [M502] au dezvoltat o metodă de identificare a nodurilor ascunse mai puţin importante şi de eliminare a acestora în timpul învăţării. Metoda se bazează pe calculul relevanţei ("relevance") ieşirii fiecărui perceptron simplu. Cei cu relevanţa cea mai scăzută sunt eliminaţi din structură.

Relevanţa ρ a fiecărui nod este calculată ca diferenţa dintre eroarea obţinută atunci când nodul este eliminat şi eroarea obţinută atunci când acesta face parte din structură. In locul calculului direct pentru fiecare nod, se poate folosi o aproximare $\hat{\rho}$ bazată pe derivata de ordinul întâi a erorii, în raport cu un termen

Scanned with OKEN Scanner

α, care măsoară importanţa perceptronului simplu în structură şi depinde de valoarea de ieşire a acestuia:

$$y_j = f\left(\sum_i w_{ij}\, \alpha_i\, y_i\right) \tag{4.198}$$

$$\hat{\rho}_i = -\left.\frac{\partial C_{\hat{i}nv}}{\partial \alpha_i}\right|_{\alpha_i = 1} \tag{4.199}$$

Funcţia cost este de tip L_1, pentru a permite o evaluare mai fină a erorilor de valoare mică:

$$C_{\hat{i}nv} = \sum_{\xi=1}^{P} \sum_{j=1}^{N} |d_j(\xi) - y_j(\xi)| \tag{4.200}$$

S–au notat cu y_i, y_j ieşirile neuronilor i, j, cu w_{ij} ponderea conexiunii dintre neuronii i şi j; f este funcţia de activare sigmoidală, iar $\hat{\rho}_i$ este estimatul relevanţei, pentru neuronul i.

Dacă $\alpha = 0$, atunci perceptronul simplu nu are nici o contribuţie în reţea; dacă $\alpha = 1$, atunci se comportă normal şi nu trebuie eliminat. Valoarea $\hat{\rho}_i$ se poate calcula prin propagarea inversă a erorii (algoritmul BKP). Intrucât se evaluează pentru $\alpha = 1$, rezultă că α este o notaţie convenţională, mai mult decât parametru ce trebuie introdus în reţea. Atunci când valoarea $\hat{\rho}_i$ este mai mică decât un prag prestabilit, nodul poate fi eliminat din reţea.

Pentru scăderea fluctuaţiilor valorii relevanţei, se poate introduce o ponderare:

$$\hat{\rho}_i(t+1) = 0.8\, \hat{\rho}_i(t) + 0.2\, \frac{\partial C_{\hat{i}nv}(t)}{\partial \alpha_i} \tag{4.201}$$

Segee şi Carter [R14], au studiat efectul metodei prezentate asupra toleranţei la defecte a reţelei, remarcând faptul că, reţeaua obţinută nu este mai sensibilă la defecte în mod semnificativ, în comparaţie cu structura iniţială, deşi are mai puţini parametri. De asemenea, se precizează că, graficul creşterii erorii în funcţie de valoarea ponderii eliminate, nu diferă semnificativ în cazul reţelei iniţiale şi finale (structură redusă) obţinută. O altă concluzie obţinută de Segee şi Carter se referă la dispersia ponderilor conexiunilor ce intră într–un nod, care este un predictor bun al relevanţei nodului. La rândul său, relevanţa nodului este

Scanned with OKEN Scanner

un predictor bun al creşterii erorii pătratice medii când ponderea de valoare maximă care intră în nod, se anulează.

- ***Metodă de câştigare a competiţiei* (II)**

Kruschke [K501] a prezentat o metodă bazată pe competiţia nodurilor ascunse prin inhibiţie laterală. Câştigă noduri din acelaşi strat, perceptronii simpli ascunşi complet inhibaţi fiind apoi eliminaţi.

4.6.4.4 Metode bazate pe regularizare

Metodele ontogenice "distructive" de reducere a structurilor au fost prezentate pentru cazul eliminării conexiunilor care nu sunt necesare. Dacă se doreşte însă eliminarea directă a nodurilor din reţea, regulile se modifică. Pentru cazul reducerii arhitecturii neurale prin scăderea ponderilor, se pot stabili rate de descreştere ("weight decay") mai mari pentru nodurile ale căror valori de ieşire sunt mici, sau pentru nodurile care primesc la intrare semnale prin conexiuni care au ponderi mici. De exemplu, se poate înlocui (4.172) cu relaţia (4.202). Valoarea parametrului ε_j se poate folosi în acest caz, pentru toate conexiunile care intră în nodul j. S-a notat cu w_{ij} ponderea conexiunii dintre nodurile i şi j.

$$\varepsilon_j = \frac{\chi \eta / 2}{\left(1 + \sum_i w_{ij}^2\right)^2} \qquad (4.202)$$

4.6.4.5 Alte metode de eliminare a câte unui nod ascuns

In această clasă se pot include metode bazate pe corelaţie, pe calculul unor norme (de exemplu, norma Frobenius), etc.

- ***Metodă bazată pe corelaţie* (I)**

Xue, Hu şi Tompkins [X2] utilizează corelaţia existentă între nodurile ascunse pentru reducerea structurii "feedforward". Setul de noduri cu independenţă maximă este păstrat. Se notează cu $\boldsymbol{Z}$ matricea cu H linii. Fiecare linie a matricii $\boldsymbol{Z}$ reprezintă ieşirile tuturor nodurilor ascunse, atunci când la intrarea reţelei a fost prezentat un anumit pattern de învăţare. Paşii algoritmului propus sunt prezentaţi în tab. 4.22.

Scanned with OKEN Scanner

Tab. 4.22 Algoritmul metodei bazate pe corelaţie (I)

❶ Se realizează o descompunere QR, cu pivotare pe coloană în matricea $\boldsymbol{Z}$, cu $\aleph$ matricea de permutare:

$$\mathbf{Z}\,\aleph = \mathbf{Q}\,\mathbf{R} \tag{4.203}$$

❷ Se determină numărul H_{final} de noduri ascunse ce trebuie păstrate. Ponderile nodurile ascunse corespunzătoare sunt primele H_{final} coloane ale matricii $\boldsymbol{Z}\ \aleph$, de dimensiuni H x H_{final}. Această matrice $\boldsymbol{Z}$ redusă se notează cu:

$$\mathbf{Z}_r = \mathbf{Q}(:,\ 1{:}\ H_{final})\ \mathbf{R}(1 : H_{final},\ 1{:}\ H_{final}) = \mathbf{Q}_1\,\mathbf{R}_1 \tag{4.204}$$

unde $\boldsymbol{Q}(:,\ 1{:}\ H_{final})$ este matricea compusă din primele H_{final} coloane ale matricii $\boldsymbol{Q}$, iar $\boldsymbol{R}(1 : H_{final},\ 1 : H_{final})$ este o matrice de dimensiuni H_{final} x H_{final}, minor principal al matricii $\boldsymbol{R}$. Matricea iniţială $\boldsymbol{Z}$ poate fi aproximată astfel:

$$\hat{\mathbf{Z}} = \mathbf{Z}_r\left[\mathbf{Z}_r^T\,\mathbf{Z}_r\right]^{-1}\mathbf{Z}_r^T\,\mathbf{Z} = \mathbf{Z}_r\left(\mathbf{R}_1^T\,\mathbf{R}_1\right)^{-1}\mathbf{R}_1^T\,\mathbf{Q}_1^T\,\mathbf{Z} = \mathbf{Z}_r\,\Theta \tag{4.205}$$

❸ Nodurile ascunse care nu pot fi păstrate sunt înlocuite cu combinaţii liniare ale ieşirilor celor H_{final} noduri ascunse păstrate. Dacă structura are un singur strat ascuns, atunci se poate deduce relaţia de modificare a ponderilor dintre acesta şi stratul de ieşire, notate cu $\boldsymbol{V}$.

Se notează cu $\hat{\boldsymbol{N}}\boldsymbol{et}$ activările nodurilor din stratul ascuns, cu pragurile $\boldsymbol{b}$. Ieşirile acestora sunt aproximate de $\hat{\boldsymbol{Z}}$ şi există relaţia:

$$\hat{\mathrm{N}}\mathrm{et} - \mathbf{b} = \hat{\mathbf{Z}}\,\mathbf{V}_0^T = \mathbf{Z}_r\,\Theta\,\mathbf{V}_0^T = \mathbf{Z}_r\,\hat{\mathbf{V}}_0^T \tag{4.206}$$

Neglijând termenii de prag, rezultă (4.207), cu matricea Θ din (4.208), pe baza relaţiei (4.205).

$$\hat{\mathbf{V}}_0 = \mathbf{V}_0\,\Theta^T \tag{4.207}$$

$$\Theta = \left(\mathbf{R}_1^T\,\mathbf{R}_1\right)^{-1}\mathbf{R}_1^T\,\mathbf{Q}_1^T\,\mathbf{Z} \tag{4.208}$$

Este important de menţionat faptul că, metoda nu este limitată la RPM total interconectate. Rezultatul poate fi aplicat şi reţelelor "feedforward" care au matricea ponderilor rară, prin completarea acestora cu noduri virtuale.

Scanned with OKEN Scanner

❒ *Metodă de aproximare a normei Frobenius* (II)

Kung şi Hu [K502] au considerat că, setul de noduri ascunse ale căror ieşiri aproximează cel mai bine nivelul de activare al reţelei originale poate să fie păstrat, pe baza unei metode de aproximare a normei Frobenius (FARM). Fiind dată matricea $\boldsymbol{Z}$, procedura FARM caută o aproximare a matricii ***Net*** (care conţine valorile *net* furnizate stratului de ieşire) prin selectarea a s coloane ale matricii $\boldsymbol{Z}$. Metoda se bazează pe aproximarea pe baza celor mai mici pătrate, în variantă recursivă (tab. 4.23).

Tab. 4.23 Algoritmul FARM

❶ Iniţializare: Se dau $\boldsymbol{Z}_{P \times H}$, q, $\boldsymbol{Net}_{P \times N}$. Iniţializează $\boldsymbol{Z}_r = 0$.

Pentru $1 \le h \le H_{final}$ DO paşii ❷, ❸, ❹, ❺.

❷ Normalizează fiecare coloană a matricii $\boldsymbol{Z}$:

$$\mathbf{Z}(:, h) = \frac{\mathbf{Z}(:, h)}{\|\mathbf{Z}(:, h)\|}, \quad 1 \le h \le H \tag{4.209}$$

❸ Selectează:

$$h^* = \arg \max_{1 \le h \le H} \|\mathbf{Z}^T(:, h)\,\mathbf{Net}\|^2 \tag{4.210}$$

❹ Creşterea matricii $\boldsymbol{Z}_r$ prin introducerea $\boldsymbol{Z}(:, h^*)$

❺ Proiectează $\boldsymbol{Z}$ în spaţiul nul al matricii $\boldsymbol{Z}(:, h^*)$:

$$\mathbf{Z} = \left(\mathbf{I} - \mathbf{Z}(:, h^*)\,\mathbf{Z}^T(:, h^*)\right)\mathbf{Z} \tag{4.211}$$

4.6.5 Comparaţie a metodelor ontogenice "distructive" de proiectare a structurilor "feedforward"

❒ *Comparaţie* (I)

Fiesler [F28] a realizat o prezentare compactă a unui număr de 32 de modele ontogenice, dintre care o parte sunt "distructive". Clasificarea este realizată considerând criteriile prezentate în par. 4.5.10.

Scanned with OKEN Scanner

❒ *Comparaţie* (II)

Propunem o clasificare succintă sub formă de tabel (tab. 4.24), a metodelor "distructive" prezentate în cap. 4.6. Numele metodei s-a indicat prin prescurtarea sa, dacă aceasta este folosită curent. Semnificaţia indicatorilor este următoarea:

X / N	Elimină con**X**iune / **N**euron
I / D	Reducere **I**n timpul învăţării sau **D**upă încheierea acesteia
R / F	Cu **R**eînvăţare sau **F**ără, după reducerea structurii
G	Generalizare
🕒	Timp
C	Complexitatea structurii
B	"Benchmark" (probleme de test)

Indicatorul generalizare se referă la valoarea erorii medii pe setul de test. Costul învăţării este direct proporţional cu timpul consumat pentru faza de învăţare şi reducere a arhitecturii neurale. Complexitatea structurii se evaluează prin numărul total al perceptronilor simpli din structura finală.

Intrucât problemele de test sunt diferite, comparaţia pe baza indicatorilor generalizare, timp şi complexitate se poate face numai calitativ.

Semnele 👍, 👎, ≈ au aceeaşi semnificaţie ca şi în paragraful anterior.

Metodele "distructive" prezintă avantaje şi dezavantaje. Este semnificativ faptul că, realizarea unei generalizări corespunzătoare, ulterior învăţării, într-un timp rezonabil, a unei structuri de o complexitate cât mai redusă, presupune realizarea unui compromis între aceşti indicatori.

Avantajele metodelor ontogenice "distructive" sunt:

❶ Reducerea complexităţii structurii neurale are drept consecinţă îmbunătăţirea generalizării. Generalizarea este mai bună pentru structurile reduse, pentru majoritatea metodelor "distructive", dar metodele bazate pe calculul senzitivităţii conduc la rezultatele cele mai bune.

❷ Prin utilizarea metodelor bazate pe regularizare, se evită fenomenul de supraînvăţare.

❸ Prin reducerea arhitecturii neurale rezultă structuri de dimensiuni mult mai reduse decât cele iniţiale, care pot rezolva probleme complexe de clasificare şi regresie.

❹ Metodele "distructive" care realizează reducerea structurii în timpul învăţării, fără a necesita o etapă de reînvăţare sunt cele mai rapide, comparativ cu celelalte tipuri de metode "distructive".

Scanned with OKEN Scanner

Tab. 4.24 Metode ontogenice "distructive" pentru proiectarea rețelelor neurale "feedforward"

Metoda "distructivă"	Autori	Anul	Nume metodă	X / N	I / D	R / F	G	🕒	C	B
Metode drastice	Thodberg	1989		X	I	R	👍	👍	👍	
	Reed	1991		X	I	R	👍	👎	👍	
Metode de grupare a ponderilor	Kruschke	1991	"Local bottleneck"	X, N	I	F	👍	👎	👍	
	Kruschke	1991	"Distributed bottleneck"	X	I	F	👍	👎	👍	
	Nowlan & Hinton	1992	"Soft weight sharing"	X	D	R	👍	👎	👍	
Metode de estimare a senzitivității	Denker & LeCun & Solla	1990	OBD	X	D	R	👍	👎	👍	Probl. contiguitate Probl. "3 călugări" Predicția petelor solare, Mackey–Glass
	Cibias & Soulié & Gallinari	1994	OCD	X	D	R	👍	👍	👍	Clasificare Predicția seriilor de timp
	Hassibi & Stork	1993	OBS	X	D	R	👍	👎	👍	Clasificare booleană

Scanned with OKEN Scanner

Tab. 4.24 Metode ontogenice "distructive" pentru proiectarea reţelelor neurale "feedforward" (cont.)

Metoda "distructivă"	Autori	Anul	Nume metodă	X / N	I / D	R / F	G	◷	C	B
Metode de estimare a senzitivităţii (cont.)	Cottrel & Girard ş.a.	1994	SSM	X	D	R	👍	👎	👍	
	Karnin	1990		X	D	R	👍	👎	👍	
Metode bazate pe regularizare	Ishikawa	1990		X	I	F	👍	≈	👍	
	Hinton	1992		X	I	F	👍	≈	👍	
	Krogh & Hertz	1991		X	I	F	👍	≈	👍	
	Weigend ş.a.	1991		X	I	F	👍	≈	👍	Predicţia seriilor de timp
	Hansen ş.a.	1994	OBD cu "weight decay"	X	D	R	👍	👎	👍	Predicţia seriilor de timp
	Hansen & Pedersen	1994	OBS cu "weight decay"	X	D	R	👍	👎	👍	Predicţia seriilor de timp
	Chauvin	1989		X	I	F	👍	≈	👍	
	Ji ş.a.	1990		X	I	F	👍	👎	👍	
	Nowlan & Hinton	1992		X	I	F	👍	👎	👍	

Scanned with OKEN Scanner

Tab. 4.24 Metode ontogenice "distructive" pentru proiectarea reţelelor neurale "feedforward" (cont.)

Metoda "distructivă"	Autori	Anul	Nume metodă	X / N	I / D	R / F	G	◷	C	B
Metode bazate pe regularizare (cont.)	MacKay			X	D	R	👍	≈	👍	Clasificare Bayes
	Yasui & Malinowski & Zurada	1995	CSDF	X, N	I	F	👍	≈	👍	Clasificarea datelor IRIS
Metode drastice	Siestma & Dow	1988		N	I	F	👍	≈	👎	Clasificare booleană
Metode de grupare a ponderilor	Mozer	1989	Skeletonizare	N	D	R	👍	≈	👎	
	Yasui & Malinowski & Zurada	1995	CSDF	N	I	F	👍	≈	👍	Clasificarea datelor IRIS
Metode bazate pe calculul senzitivităţii	Mozer & Smolensky	1989	Skeletonizare	N	D	R	👍	≈	👎	
	Kruschke	1989	Câştigarea competiţiei	N	I	F	👍	≈	👍	
Metode bazate pe regularizare	Krogh & Hertz	1991		N	I	F	👍	≈	👍	
Alte metode	Xue & Hu & Tompkins	1991	Corelaţie	N	I	F	👍	👎	👍	
	Kung & Hu	1991	FARM	N	I	F	👍	👎	👍	

Scanned with OKEN Scanner

Dezavantajele metodelor ontogenice "distructive" sunt:

❶ Alegerea dimensiunii iniţiale a structurii este problematică.

❷ Metodele drastice implică alegerea empirică a unor parametri, care se realizează prin încercări.

❸ Intrucât în majoritatea intervalului de învăţare, reţeaua are dimensiuni mari, metodele "distructive" sunt în general consumatoare de timp. Cu toate acestea, între diferite tipuri de metode "distructive" se pot remarca diferenţe din acest punct de vedere.

Metodele bazate pe calculul senzitivităţii implică de obicei realizarea reducerii structurii după încheierea etapei de învăţare, precum şi o etapă ulterioară de reînvăţare. De aceea, timpul de calcul este ridicat.

Metodele bazate pe regularizare, deşi realizează descreşterea structurii în timpul învăţării şi nu implică nici o etapă de reînvăţare, nu realizează totuşi o economisire a timpului, datorită utilizării funcţiei cost regularizată, al cărei calcul este consumator de timp.

❹ Metodele "distructive" care evaluează modificarea erorii la ştergerea unei conexiuni / nod din structură, aproximează această modificare prin dezvoltarea Taylor (cu termeni de ordinul întâi sau doi). Dacă se consideră că matricea hessiană a funcţiei eroare este diagonală, metoda este valabilă numai dacă se elimină câte o singură conexiune. Pe de altă parte, prin această ipoteză se introduc erori. Renunţarea la ipoteza diagonalităţii matricii hessiene este o variantă mai generală, care elimină o parte dintre erori, dar timpul de calcul creşte.

❺ Metodele "distructive" bazate pe regularizare presupun stabilirea balanţei dintre termenul de eroare şi cel de penalizare, prin alegerea parametrilor de regularizare. Această problemă se poate elimina prin considerarea regularizării adaptive. Totuşi, timpul de învăţare creşte datorită funcţiei cost regularizate, iar termenul de penalizare tinde să creeze puncte de minim local suplimentare.

❻ Deşi complexitatea structurii scade cu reducerea reţelei, micşorarea dimensiunilor arhitecturii neurale nu este totdeauna benefică [H26]. In cazul descreşterii prin regularizare adaptivă şi prin regularizare de tip Bayes, eroarea de generalizare poate să crească dacă presupunerile ce stau la baza alegerii unui anumit termen de regularizare nu corespund mediului real (datelor). Prin compararea schemelor de regularizare se ajunge la concluzia că, eficienţa schemelor adaptive de regularizare, deci şi cea a metodelor de reducere a structurii, depinde în mod critic de distribuţia valorilor dorite.

❼ Problemele de test sunt diferite, de aceea evaluarea comparativă a performanţelor tuturor metodelor ontogenice "distructive" este dificilă.

❽ Nici una dintre metodele prezentate nu este optimală.

Scanned with OKEN Scanner

4.7 METODE ONTOGENICE MIXTE

Metodele ontogenice mixte de proiectare a structurilor neurale "feedforward" combină elemente ce aparţin atât tehnicilor constructive, cât şi celor "distructive". Scopul este obţinerea unor performanţe mai bune, întrucât nici una dintre metodele anterioare nu este optimală.

4.7.1 Clasificarea metodelor ontogenice mixte

Există un număr redus de metode mixte de proiectare a structurilor neurale. Acestea pot fi clasificate în două grupe, prezentate în tab. 4.25. Metodele secvenţiale au la bază algoritmi în care se pot distinge net, etapele constructivă şi "distructivă", în timp ce metodele nesecvenţiale nu conţin faze delimitate strict, de construire / reducere a structurii.

Tab. 4.25 Clasificarea metodelor ontogenice mixte

Metode ontogenice mixte	■ Metode secvenţiale
	■ Metode nesecvenţiale

4.7.2 Metode secvenţiale (constructive – "distructive")

In componenţa metodele secvenţiale se pot recunoaşte: ❶ etapa constructivă şi ❷ etapa "distructivă". Algoritmii ce caracterizează fiecare dintre aceste etape au fost prezentaţi în par. 4.5 şi 4.6.

❒ ***Metodă secvenţială* (I)**

Hirose, Yamashita şi Hijiya [H502, N12] au propus un algoritm similar metodei constructive sugerate de Ash (par. 4.5), introducând în plus o etapă "distructivă". Imediat după convergenţa reţelei, ultimul nod adăugat este eliminat, iar reţeaua învaţă din nou.

❒ ***Metodă secvenţială* (II)**

Metoda lui Ivakhnenko [I500] se bazează pe adăugarea câte unui strat şi eliminarea câte unui nod.

Scanned with OKEN Scanner

❒ ***Metodă secvenţială* (III)**

Barron [B501] a propus o metodă, care adaugă câte un strat şi elimină câte un nod.

❒ ***Metodă secvenţială* (IV)**

Metoda lui Fritzke [F501], utilă pentru structuri de tip hartă auto–organizabilă, realizează atât creşterea, cât şi descreşterea reţelei la nivel de perceptron simplu.

❒ ***Metodă secvenţială* (V)**

In lucrarea [D202] am propus o metodă constructivă – "distructivă" pentru proiectarea filtrelor neurale. Etapa constructivă se bazează pe algoritmul CASCOR cu o structură iniţială modificată, iar cea de reducere a reţelei rezultate foloseşte metoda OBD.

❶ Etapa constructivă, folosind algoritmul CASCOR

Funcţia cost utilizată este norma L_2 a erorii (4.212), care este o funcţie aditivă, deci se poate scrie sub forma (4.136), în care $C(\xi)$ are expresia (3.62). La intrarea reţelei se prezintă setul de patternuri de învăţare. Valoarea ieşirii oricărui nod de ieşire j, cu $1 \le j \le N$, este dată de relaţia (4.213).

$$C_{inv} = \frac{1}{2} \sum_{j=1}^{N} \sum_{\xi=1}^{P} \left(y_j(\xi) - d_j(\xi) \right)^2 \qquad (4.212)$$

$$y_j(\xi) = \tanh\left(\sum_{i=1}^{M+H} w_{ij}\, x_i(\xi) \right) + w_0 \qquad (4.213)$$

Ponderile conexiunilor de la nodurile de intrare la nodul ascuns introdus sunt modificate, pentru maximizarea corelaţiei S_h dintre ieşirea acestuia $z_h(\xi)$ şi funcţia cost (eroare) de învăţare $C_j(\xi)$ în nodul de ieşire j, conform relaţiei (4.74). In acest scop, se calculează derivata corelaţiei S_h în raport cu fiecare pondere de intrare în nodul candidat. Pe de altă parte, trebuie minimizată funcţia cost prin metoda gradientului negativ, cu relaţiile (4.76) – (4.80).

❷ Etapa de reducere a structurii, folosind metoda OBD

Metoda OBD (par. 4.6) se bazează pe eliminarea conexiunilor care au

Scanned with OKEN Scanner

salienţele cele mai scăzute. Salienţa este evaluată pe baza creşterii erorii δC la ieşirea structurii, atunci când vectorul de ponderi $\mathbf{w}$ este perturbat, prin eliminarea unei componente. Aproximarea creşterii erorii se realizează conform relaţiilor (4.131), (4.132), de unde rezultă că salienţa fiecărei ponderi este dată de (4.134), respectiv (4.135). Pentru funcţia cost (4.136), folosind aproximaţiile pe care se bazează metoda OBD, se obţine salienţa ponderii w_q, cu $1 \le q \le \Pi$ (relaţia (4.139)).

Pentru reţeaua piramidală obţinută în pasul ❶, cu M intrări, H straturi ascunse, fiecare cu câte un singur neuron (un număr total de H noduri ascunse), şi N noduri de ieşire, se pot obţine expresiile explicite ale salienţelor ponderilor. Funcţia globală implementată de reţea este:

$$F(\mathbf{w}; \mathbf{x}(\xi)) = \left[F_1(\mathbf{w}; \mathbf{x}(\xi)), \ldots, F_j(\mathbf{w}; \mathbf{x}(\xi)), \ldots, F_N(\mathbf{w}; \mathbf{x}(\xi))\right] \tag{4.214}$$

unde:

$$F_j(\mathbf{w}; \mathbf{x}(\xi)) = \sum_{i=1}^{M+H} \tanh\left(w_{ij}\, x_i(\xi)\right) + w_0 \tag{4.215}$$

Derivata sa în raport cu ponderea w_{ij} a conexiunii dintre perceptronii simpli i şi j este dată de relaţia (4.216), iar salienţa ponderii este (4.217), unde $1 \le i, m \le M + H$ şi $1 \le j \le N$.

$$\frac{\partial F_j(\mathbf{w}; \mathbf{x}(\xi))}{\partial w_{ij}} = \left[1 - \tanh^2\left(\sum_{m=0}^{M+H} w_{mj}\, x_m(\xi)\right)\right] x_i(\xi) \tag{4.216}$$

$$s_{ij} = w_{ij}^2 \frac{1}{2P} \sum_{\xi=1}^{P} \left[1 - \tanh^2\left(\sum_{m=1}^{M+H} w_{mj}\, x_m(\xi)\right)\right]^2 \tag{4.217}$$

Pe baza valorilor salienţelor, se construieşte o ierarhie a importanţei ponderilor. Conexiunile care au valorile cele mai mici ale salienţei sunt eliminate, iar reţeaua parcurge o etapă de reînvăţare.

Problema de studiu abordată în lucrarea [D202], a fost refacerea imaginii Lenna[20], de dimensiuni 255 x 255 pixeli, cu zgomot gaussian. In fig. 4.18 este

[20]Denumirea generică de "restoration" include operaţiile de filtrare / estimare a semnalului, în sensul "refacerii" semnalului iniţial. Formulată ca o problemă de estimare, refacerea imaginii presupune estimarea imaginii iniţiale $x(u)$ atunci când se cunoaşte o replică distorsionată $x_{zg}(u)$ a acesteia: $x_{zg} = D\,x + e$, cu D matricea de distorsionare şi e vectorul erorilor.
Când nu se cunoaşte matricea de distorsiune şi nici distribuţia zgomotului, este utilă aplicarea unei reţele perceptron multistrat pentru refacerea semnalului, conform fig. 4.18. Avantajul esenţial este acela că, reţeaua neurală estimează imaginea iniţială, valoarea sa de ieşire fiind estimatul condiţionat:

Scanned with OKEN Scanner

prezentată schematic problema de studiu, structura reţelei neurale fiind considerată pe rând, fixă, respectiv variabilă. Imaginea corectă de intrare, precum şi imaginea dorită (corectă), de dimensiuni Q x Q, au fost citite cu ajutorul unei ferestre de dimensiuni W x W pixeli, care s-a deplasat cu pasul $W/2$ pixeli spre dreapta. S-a notat cu u mulţimea perechilor $\{u = (u_1, u_2), 1 \le u_1, u_2 \le Q\}$.

Vectorul de intrare pentru RNA perceptron multistrat conţine valorile nivelelor de gri ale pixelilor din fereastra centrată pe pixelul (u_1, u_2). In timpul etapei de învăţare, ieşirile dorite sunt nivelele de gri ale pixelilor din imaginea iniţială. După învăţare, performanţele de generalizare ale RPM sunt evaluate pe imaginea de test, cu zgomot gaussian. Criteriile de evaluare a performanţelor au fost considerate eroarea pătratică medie (EPM) (4.218), eroarea pătratică medie normată (EPMN) (4.219) şi raportul semnal – zgomot la vârf (PSNR, "peak signal to noise ratio") (4.220) [P28].

$$EPM = \frac{1}{Q^2} \sum_{u_1=1}^{Q} \sum_{u_2=1}^{Q} \left(\hat{x}(u_1, u_2) - x(u_1, u_2)\right)^2 \qquad (4.218)$$

$$EPMN = \frac{1}{P\ Q^2} \sum_{u_1=1}^{Q} \sum_{u_2=1}^{Q} \left(\hat{x}(u_1, u_2) - x(u_1, u_2)\right)^2 \qquad (4.219)$$

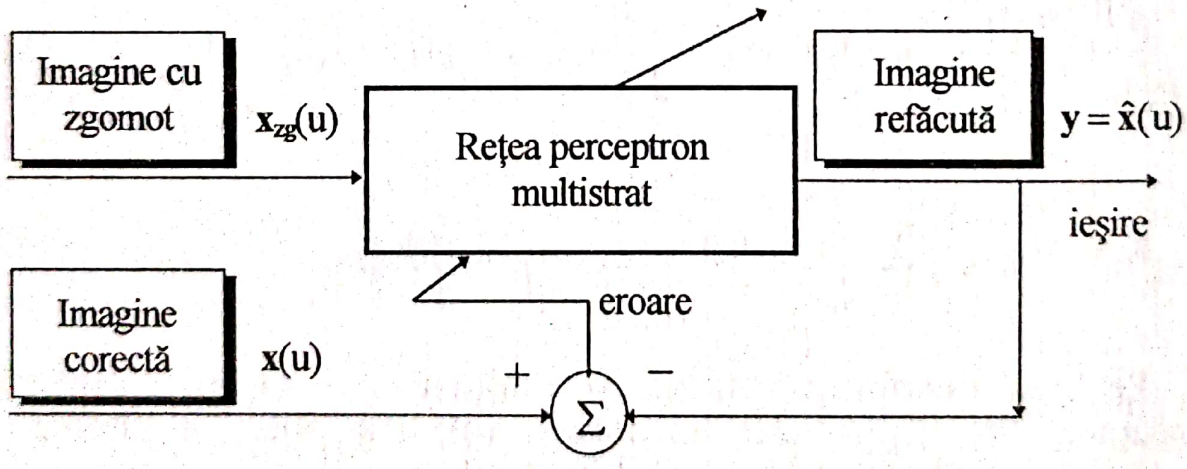


Fig. 4.18 Problema de studiu: refacerea imaginii folosind RPM

$y = E\{x(u_1, u_2) \mid x_{zg,\ fereastră}(u_1, u_2)\}$
unde s-a notat cu $x_{zg,\ fereastră}(u_1, u_2)$ setul de eşantioane de intrare din imaginea zgomotoasă, aparţinând ferestrei centrate pe pixelul (u_1, u_2). Dacă se estimează mai mult de un pixel ($N > 1$), atunci relaţia anterioară devine:
$y_j = E\{x(u_1, u_2) \mid x_{zg,\ fereastră}(u_1, u_2)\},\ 1 \le j \le N$
Dacă componentele vectorului y_j sunt independente, şi uniform distribuite, atunci relaţia este valabilă pentru fiecare componentă în parte:
$y_j(u_1, u_2) = E\{x(u_1, u_2) \mid x_{zg,\ fereastră}(u_1, u_2)\},\ 1 \le j \le N$

Scanned with OKEN Scanner

$$\text{PSNR} = 10\log_{10}\frac{(255)^2}{\frac{1}{Q^2}\sum_{u_1=1}^{Q}\sum_{u_2=1}^{Q}\left(\hat{x}(u_1,u_2)-x(u_1,u_2)\right)^2}\,[\text{dB}] =$$

$$= 10\log_{10}\frac{(255)^2}{\text{EPM}}\,[\text{dB}] \qquad (4.220)$$

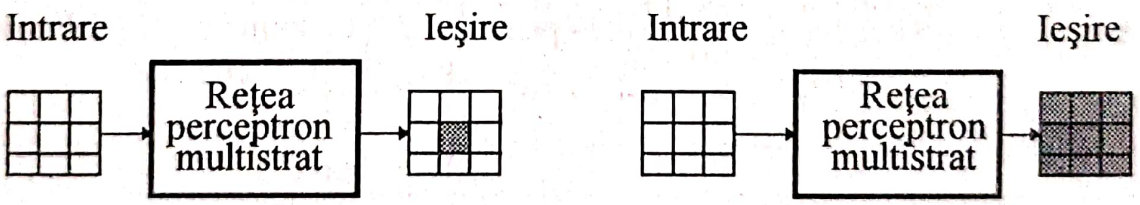
unde Q x Q este numărul pixelilor din imagine, $x(u_1, u_2)$ este valoarea intensităţii pixelului corect, iar $\hat{x}(u_1, u_2)$ este valoarea intensităţii pixelului obţinut la ieşirea reţelei.

■ Cazul structurii fixe

Structura reţelei a fost *M–H–N*, respectiv 9–5–1 şi 9–5–9, unde $M = N = W$ x $W = 9$.

Experimentul 1: In faza de învăţare, vectorii de intrare rezultaţi în urma citirii imaginii printr-o fereastră 3 x 3 pixeli, sunt prezentaţi la intrarea reţelei neurale. Ieşirea dorită este valoarea pixelului din centrul ferestrei. Dimensiunile reţelei sunt 9–5–1.

Experimentul 2: Vectorii de intrare obţinuţi prin citirea imaginii printr-o fereastră 3 x 3 pixeli, sunt prezentaţi la intrarea reţelei neurale. Ieşirile dorite sunt aceleaşi cu valorile de intrare. Reţeaua are dimensiunile 9–5–9.



Experimentul 1 Experimentul 2

Fig. 4.19 Experimentele 1 şi 2, pentru cazul structurii fixe, în scopul alegerii dimensiunii stratului de ieşire al RPM

Pentru ambele experimente, ponderile iniţiale au fost aleatoare, în intervalul (0,1), rata de învăţare egală cu 0.5, iar numărul de epoci egal cu 1000. Rezultatele experimentale pentru etapa de învăţare sunt prezentate în tab. 4.26. Datorită numărului diferit de parametri (56 pentru RPM în experimentul 1 şi respectiv, 104 pentru RPM în experimentul 2), eroarea de învăţare după 1000 de

Scanned with OKEN Scanner

epoci este mai mare în cazul experimentului 2. S-a continuat învăţarea în acest caz, până la obţinerea unei valori a erorii de învăţare aproximativ egală cu cea obţinută în primul caz.

Rezultatele etapei de test sunt prezentate în tab. 4.27, pentru valorile deviaţiei standard a zgomotului gaussian σ = 20, 30 şi 40. Erorile de test sunt mai ridicate în cazul primului experiment, pentru toate valorile σ luate în considerare. Diferenţele între rezultatele celor două experimente, din punct de vedere al valorilor PSNR, variază între minimum 1.24 dB şi maximum 1.42 dB. Rezultatele obţinute sugerează posibilitatea alegerii dimensiunii de ieşire a reţelei, $N > 1$. De asemenea, este important faptul că, în cazul experimentului 2, se poate consideră că învăţarea este supervizată chiar de semnalul de intrare deci, în ansamblu, este nesupervizată.

Comparaţia cu rezultatele obţinute de filtrul optimal Wiener (fereastră 3 x 3) indică valori PSNR mai ridicate pentru structura neurală, în ambele experimente şi pentru toate valorile σ, cu excepţia cazului σ= 20. Diferenţa între rezultele obţinute de RNA şi structura Wiener, din punct de vedere al valorilor PSNR variază între minimum 0.04 dB şi maximum 1.28 dB în favoarea celei din urmă (pentru σ = 20), între minimum 1.19 dB şi maximum 2.61 dB în favoarea RNA (pentru σ= 30), între minimum 3.03 dB şi maximum 4.31 dB în favoarea RNA (pentru σ = 40). Se constată utilitatea evidentă a aplicării structurii neurale, pe măsură ce nivelul de zgomot în imaginea de intrare creşte.

Tab. 4.26 Erorile după 1000 de epoci de învăţare, pentru experimentele 1 şi 2 Imaginea Lenna, 255 x 255 pixeli, fereastră 3 x 3 pixeli, ponderi iniţiale aleatoare în intervalul (0, 1), rata de învăţare η = 0.5

Exp.	Structura reţelei	Eroarea pătratică medie normată (EPMN) după 1000 de epoci de învăţare
1	9 – 5 – 1	0.000851
2	9 – 5 – 9	0.006228

Tab. 4.27 Rezultate experimentale (EPMN şi PSNR) pentru imaginea de test Lenna, 255 x 255 pixeli, cu zgomot gaussian. Comparaţie cu rezultatele obţinute de filtrul optimal Wiener

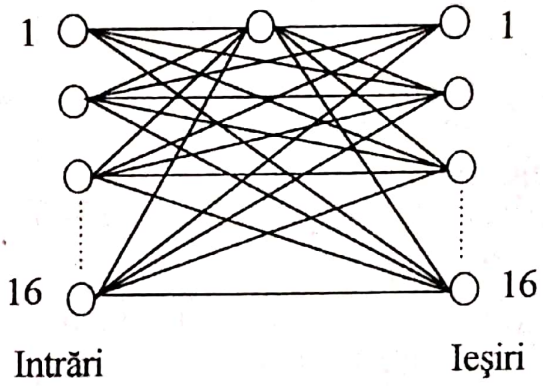
σ	Experimentul 1		Experimentul 2		Wiener	
	EPMN	PSNR [dB]	EPMN	PSNR [dB]	EPMN	PSNR [dB]
20	0.002032	75.05	0.001527	76.29	0.001512	76.33
30	0.002191	74.72	0.001581	76.14	0.002879	73.53
40	0.002336	74.41	0.001754	75.69	0.004729	71.38

Scanned with OKEN Scanner

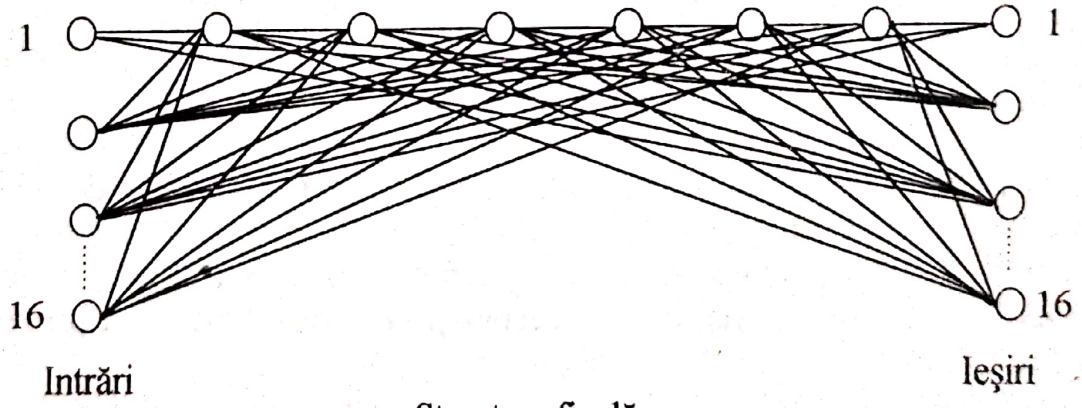
■ Cazul structurii variabile

Imaginea de intrare şi imaginea de ieşire dorită au fost citite cu ajutorul unei ferestre de dimensiune W x W = 4 x 4 pixeli, care s-a deplasat spre dreapta cu câte 2 pixeli. Structura iniţială a reţelei a fost în acest caz 16–1–16 (structură modificată faţă de varianta clasică CASCOR). Panta funcţiei de activare sigmoidală a nodului a fost 0.5, iar criteriul de evaluare a fost considerat EPM.

Structura finală obţinută după pasul ❶ al algoritmului propus este 16–1–1–1–1–1–1–16 (fig. 4.20). Erorile obţinute după etapa de învăţare cu algoritmul CASCOR, erorile de test pentru imagini cu zgomot gaussian cu deviaţia standard a) $\sigma = 20$; b) $\sigma = 30$; c) $\sigma = 40$, precum şi comparaţia cu cazul structurii fixe de dimensiuni 16–10–16, care învaţă cu algoritmul standard BKP şi cu cazul filtrării Wiener (fereastră 4 x 4), sunt prezentate în tab. 4.28.



Structura iniţială



Structura finală

Fig. 4.20 Structura iniţială şi structura finală în pasul ❶ (CASCOR)

Structura variabilă învaţă cu algoritmul propus, într-un număr de epoci de 1.92 ori mai mic decât cel al reţelei cu dimensiuni fixe. Valorile PSNR în etapa de test sunt mai ridicate pentru reţeaua propusă pentru toate cazurile luate în considerare, cu minimum 1.5 dB (pentru $\sigma = 30$) şi maximum 1.75 dB (pentru

Scanned with OKEN Scanner

$\sigma = 40$). Comparativ cu reţeaua proiectată, structura Wiener obţine rezultate mai bune din punct de vedere al valorilor PSNR, cu excepţia cazului $\sigma = 40$, când reţeaua obţine cu 0.03 dB mai mult.

Valorile mai scăzute ale raportului semnal – zgomot obţinute pentru structura variabilă, faţă de cazul structurii fixe considerată anterior, se datorează mai ales alegerii dimensiunii mai mari a ferestrei (4 x 4, faţă de 3 x 3).

Tab. 4.28 Rezultatele în etapa de învăţare şi de test, pentru imaginea Lenna, 255 x 255 pixeli, în cazurile structurii variabile cu algoritmul CASCOR, structurii fixe cu algoritmul BKP şi filtrului Wiener. Valorile erorilor şi PSNR sunt măsurate la ieşirea reţelei

Etapa de învăţare / test	Parametri şi rezultate		BKP, structură fixă	CASCOR	Wiener
Etapa de învăţare	Număr epoci		77	40	—
	Rata de învăţare η		0.3	0.4	—
	EPM de învăţare		0.01870	0.01401	—
	PSNR [dB]		65.412	66.666	—
Etapa de test	$\sigma = 20$	EPM de test	0.02010	0.01421	0.00570
		PSNR [dB]	65.098	66.604	70.559
	$\sigma = 30$	EPM de test	0.02075	0.01425	0.00960
		PSNR [dB]	64.960	66.592	68.327
	$\sigma = 40$	EPM de test	0.02150	0.01435	0.01450
		PSNR [dB]	64.806	66.562	66.532

In pasul ❷ al algoritmului, s-au eliminat 5% din numărul total de ponderi prin metoda OBD. Eroarea de test a reţelei după reducerea structurii a fost de 0.1489. Reţeaua a parcurs etape de reînvăţare. Imaginea (de test) de intrare şi cea obţinută la ieşire în acest caz, precum şi imaginea dorită sunt prezentate în lucrarea [D202].

Evaluarea vizuală a calităţii imaginii obţinute indică faptul că, rezultatul obţinut este bun, numărul parametrilor structurii neurale rezultate fiind de 1.8 ori mai scăzut decât cel al reţelei de dimensiuni fixe (tab. 4.29). Numărul parametrilor reţelei neurale cu structură fixă şi interconectare totală este dat de relaţia (4.221). Pentru structura construită pe baza algoritmului propus, numărul parametrilor se calculează cu expresia (4.222), unde ε este procentul de ponderi care sunt şterse în etapa ❷. In ambele cazuri, s-a considerat şi nodul suplimentar introdus pentru prag (par. 3.1).

Scanned with OKEN Scanner

$$\Pi = (M + 1)\, H + (H + 1)\, N \tag{4.221}$$

$$\Pi = [(M + 1)\,(H - 1) + (H + 1)\, N + (H - 1)]\,(1 - \varepsilon) \tag{4.222}$$

Numărul total al operaţiilor de "înmulţire + adunare" pentru o epocă de învăţare, este dat de relaţia (4.223) pentru structura fixă şi de (4.224) pentru structura variabilă (etapa ❶, cu algoritmul CASCOR). Numărul operaţiilor de "înmulţire + adunare" în timpul etapei de test este dat de relaţia (4.223) pentru structura fixă şi de (4.225) pentru cea proiectată. Având în vedere faptul că, numărul nodurilor ascunse obţinute în etapa ❶ este redus, $H^2 << P$ şi timpul necesar etapei de învăţare este mult mai scurt pentru reţeaua proiectată cu metoda propusă.

$$op_{inv} \cong P\,(M + N)\, H + (M + 1)H + (H + 1)N = (M + N)(P + 1)\, H + H + N \cong$$

$$\cong (M + N)\, P\, H \tag{4.223}$$

$$op_{inv} \cong 2 \sum_{h=1}^{H} \left[(M + 1)(h - 1) + (h + 1)\, N + (h - 1)\right] + P =$$

$$= (M + N + 2)\, P\, H\, (H + 1) \cong (M + N)\, H^2 + P \tag{4.224}$$

$$op_{test} \cong [(M + 1)\,(H - 1) + (H + 1)\, N + H - 1]\, P\,(1 - \varepsilon) \cong$$

$$\cong [(M + N)\, H - (M - N - H + 2)]\, P\,(1 - \varepsilon) \tag{4.225}$$

Tab. 4.29 Comparaţie între structura complet interconectată şi cea parţial interconectată obţinută, din punct de vedere al numărului parametrilor şi numărului operaţiilor de "înmulţire + adunare"

Criteriul de comparaţie		Structura fixă 16–10–16, cu algoritmul BKP	Structura complet interconectată, rezultată după pasul ❶	Structura parţial interconectată rezultată după pasul ❷
Nr. parametrilor (ponderi, praguri)		346	202	192
Nr. operaţiilor de "înmulţire + adunare"	Învăţare	$O((M+N)\, P\, H)$	$O((M+N)H^2 + P)$	—
	Test	$O((M+N)\, P\, H)$	—	$O((M+N)H\, P(1-\varepsilon))$

Timpul consumat în etapa de test este comparabil pentru cele două tipuri de structuri, diferenţa crescând odată cu valoarea parametrului ε. Pentru $P = 16256$, şi $\varepsilon = 5\%$, timpul necesar testării cu structura variabilă este de 1.72 ori mai scurt comparativ cu cel necesar structurii fixe.

In concluzie, structura neurală rezultată are performanţe mai bune decât structura fixă, în cazul refacerii imaginilor cu zgomot gaussian. De asemenea, numărul conexiunilor structurii finale este mai redus decât în cazul RPM total interconectată, iar timpul de învăţare şi de test mai scurt. Cu toate acestea, având în vedere valoarea raportului semnal – zgomot (PSNR) la ieşire după faza de învăţare, este evident că este necesară creşterea structurii în continuare, pentru a micşora valoarea erorii. O posibilitate de îmbunătăţire a performanţelor structurii proiectate, comparativ cu filtrul Wiener, este reducerea dimensiunilor ferestrei.

Pentru cazul structurii variabile, eroarea de test după reducerea cu metoda OBD şi reînvăţare este mai ridicată decât pentru reţeaua obţinută cu algoritmul CASCOR fără reducerea ulterioară a structurii, dar numărul ponderilor eliminate poate fi redus, pentru a obţine o valoare dorită a erorii de test.

Un dezavantaj se referă la faptul că, reţeaua finală nu este simetrică. Această problemă este comună aproape tuturor metodelor ontogenice, excepţie făcând o parte dintre cele care sunt aplicate problemelor de clasificare binară. Este esenţială însă, găsirea unor metode care să ia în considerare constrângeri de simetrie, în scopul implementării mai simple a structurii rezultate prin tehnici accesibile.

❒ *Metoda G – S ("Generating – Shrinking Algorithm")* (VI)

Metoda "Generating – Shrinking" (G – S) propusă de Chen, Thomas şi Nixon [C23] construieşte o RPM cu trei straturi active, pentru a obţine orice clasificare în spaţiul euclidian n – dimensional, şi apoi elimină noduri din structură. Pentru aprecierea performanţelor de clasificare ale algoritmului, se foloseşte indicatorul rată de clasificare corectă (RCC) ("correct classification rate"), definit ca:

$$RCC = \frac{P_{corect}}{P} \times 100 \ [\%] \qquad (4.226)$$

unde P_{corect} este numărul patternurilor clasificate corect, iar P este numărul total de patternuri.

Reţeaua "feedforward" este formată din trei straturi, în afara celui de intrare. Primul strat ascuns este compus din n_{L1} perceptroni simpli liniari, cel de-al doilea strat conţine n_{L2} noduri sigma – pi, iar stratul de ieşire este alcătuit din n_{L3} perceptroni simpli liniari cu prag.

Scanned with OKEN Scanner

Dimensiunea straturilor ascunse variază dinamic, iar numărul nodurilor din stratul de ieşire N este egal cu numărul claselor n_C din problema de clasificare.

Se notează cu:

- z_h ieşirea perceptronilor simpli din stratul ascuns $L1$
- y_m ieşirea perceptronilor simpli din stratul ascuns $L2$
- o_j ieşirile perceptronilor simpli din stratul de ieşire $L3$
- x'_i componenta indice i din patternul de intrare extins x'
- w ponderile dintre stratul de intrare şi stratul $L1$
- v ponderile dintre straturile $L1$ şi $L2$
- g ponderile dintre straturile $L2$ şi $L3$.
- n_C numărul claselor

Relaţiile intrare – ieşire pentru fiecare strat sunt descrise de expresiile (4.227) – (4.230) [C23].

$$z_h = \sum_{i=1}^{M+1} w_{ih}\, x'_i \qquad \text{pentru } 1 \le h \le n_{L1} \qquad (4.227)$$

$$y_m = \prod_{h=1,\, h\neq m}^{n_{L1}} f\left(v_{hm}\, z_h + v_{mm}\, z_m\right) \qquad \text{pentru } 1 \le m \le n_{L2} \qquad (4.228)$$

$$o_j = \operatorname{sgn}\left(\sum_{m=1}^{n_{L2}} g_{mj}\, y_m + b_j\right) \qquad \text{pentru } 1 \le j \le n_{L3} \qquad (4.229)$$

$$b_j = -0.5,\ \forall j, \text{ cu } 1 \le j \le n_{L3} \qquad (4.230)$$

unde *sgn* este funcţia semn unipolară.

La intrarea reţelei se prezintă un pattern M – dimensional, $x \in \mathcal{R}^M$, care trebuie clasificat, precum şi un număr de referinţă $r \in \mathcal{R}$. Impreună, x şi r formează un vector numit pattern de intrare extins, de dimensiune $M + 1$, notat $x' = (x, r) \in \mathcal{R}^{M+1}$. Fiecare valoare +1 a unei componente j din vectorul de ieşire implică apartenenţa vectorului de intrare la clasa j, $1 \le j \le n_C$.

Etapele algoritmului G – S sunt prezentate în tab. 4.30.

Teorema 4.3 Reţeaua construită în etapa ❶ a metodei G – S garantează clasificarea fiecărui pattern din setul de învăţare.

Semnificaţia numărului de referinţă r este precizată de Chen, Thomas şi Nixon: când $r \to 0$, toate hiperplanele de clasificare trec prin origine şi formează

unghiuri egale între ele atunci când separă patternurile. Când $r \to \infty$, hiperplanele separă patternurile de învăţare astfel încât să existe distanţe egale între ele (funcţionează ca un clasificator "nearest – neighbour").

Tab. 4.30 Algoritmul G – S

❶ Etapa constructivă

① Se construieşte reţeaua cu dimensiunile:
$n_{L1} = n_{L2} = P, \quad n_{L3} = n_C$

② Se alege numărul de referinţă r după cum se va preciza în continuare.

③ Pentru $1 \le \psi \le P, 1 \le i \le M+1,$

$$w_{i\psi} = \begin{cases} c\, x_i(\psi) & \text{dacă } 1 \le i \le M \\ c\, r & \text{dacă } i = M+1 \end{cases} \qquad (4.231)$$

unde $x_i(\psi)$ este componenta indice i din patternul de învăţare indice ψ, notat $x(\psi)$, iar c este:

$$c = \frac{1}{\sqrt{r^2 + \sum_{i=1}^{M} \left(x_i(\psi)\right)^2}} \qquad (4.232)$$

④ Pentru $1 \le \psi \le P, 1 \le \xi \le P,$

$$v_{\psi\xi} = \begin{cases} 1 & \text{dacă } \psi = \xi \\ -1 & \text{dacă } \psi \ne \xi \end{cases} \qquad (4.233)$$

⑤ Pentru $1 \le \xi \le P, 1 \le j \le n_C,$

$$g_{\xi j} = \begin{cases} 1 & \text{dacă } j = o\,|_{x(\xi)} \text{ (egal cu ieşirea reţelei pentru patternul } x(\xi)) \\ 0 & \text{altfel} \end{cases} \qquad (4.234)$$

❷ Etapa de reducere a reţelei rezultate în ❶ este prezentată în organigrama din fig. 4.21.

Scanned with OKEN Scanner

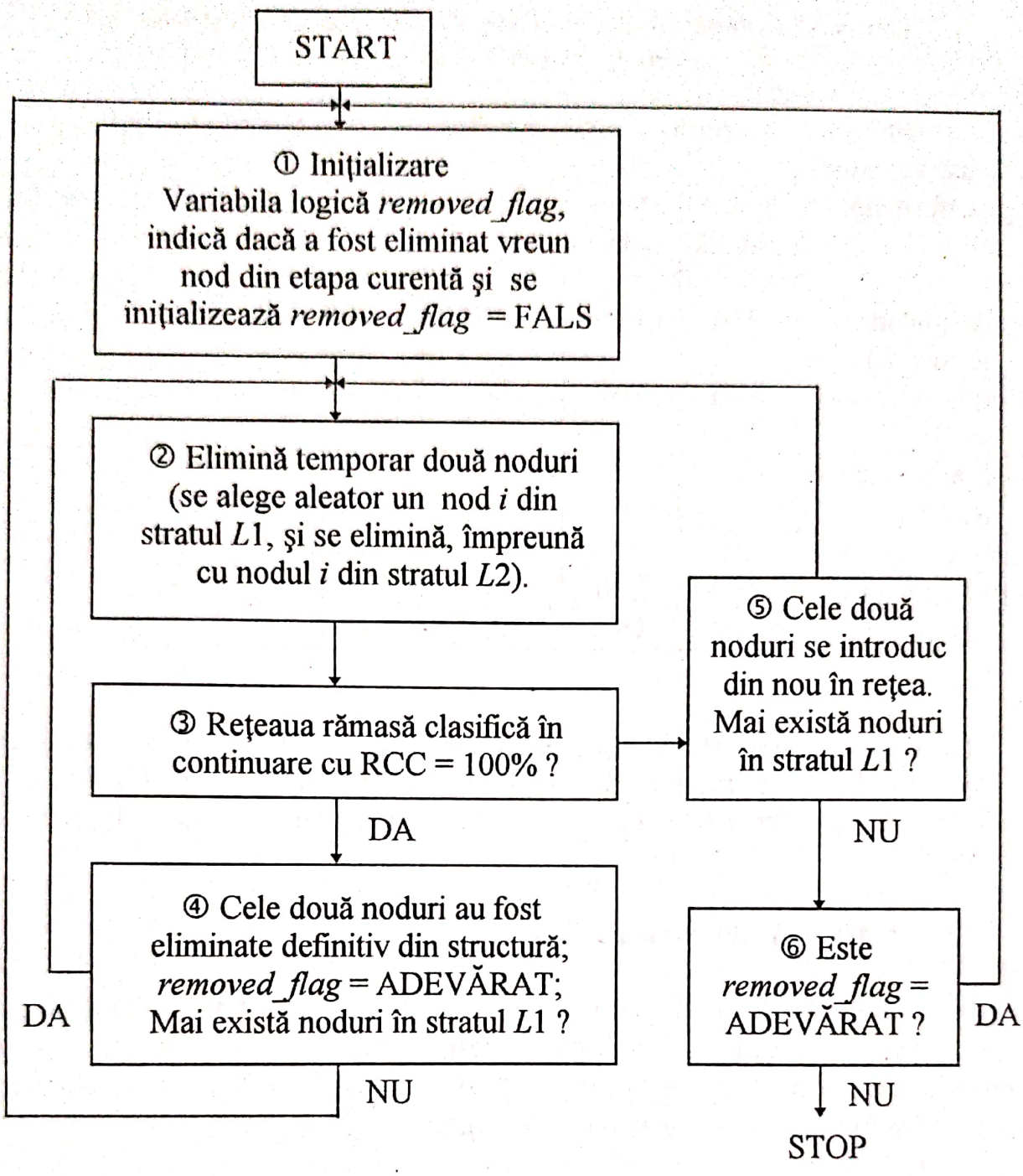


Fig. 4.21 Organigrama etapei de reducere a reţelei, în metoda G – S

Metoda a fost aplicată pentru rezolvarea problemei de clasificare "două spirale" şi pentru clasificarea texturilor. Rezultatele, precum şi comparaţia cu BKP sunt prezentate în tab. 4.31. Algoritmul G – S este mai rapid (de câteva sute de ori) şi generalizarea obţinută este mai bună decât pentru algoritmul BKP cu structură fixă. Numărul nodurilor se reduce semnificativ. Dezavantajul principal al metodei este faptul că, implică un volum mare de calcul: în etapa constructivă, acesta este proporţional cu numărul patternurilor de învăţare şi cu dimensiunea acestora.

Scanned with OKEN Scanner

Tab. 4.31 Comparaţia rezultatelor algoritmilor G – S şi BKP, pentru problema "două spirale"

Dimensiunile reţelei rezultate şi performanţe	Algoritmul G – S, $r = 500$			Algoritmul BKP		
	Reţea iniţială	Reţea redusă a)	Reţea redusă b)	Caz I	Caz II	Caz III
Noduri în stratul ascuns $L1$	500	100	36	10	10	20
Noduri în stratul ascuns $L2$	500	100	36	10	10	20
Noduri în stratul de ieşire $L3$	2	2	2	2	2	2
Timp necesar [s]	0.2	123	176	3780	11590	10180
RCC învăţare	100	100	100	97.8	91.8	100
RCC test	100	95.8	91.4	85.2	77.7	89.6

Deşi acest număr este finit şi poate fi cunoscut apriori, de obicei este mare. In faza de reducere a structurii, volumul de calcul variază cu numărul nodurilor eliminate, cazul cel mai defavorabil fiind acela în care fiecare nod este eliminat într-un pas.

❒ ***Metodă secvenţială* (VII)**

Metoda propusă de Nahban şi Zomaya [N12] a fost prezentată în par. 4.5.6. Având în vedere faptul că operaţiile efectuate sunt crează / şterge element, unde elementul este neuron / strat, această metodă poate fi inclusă şi în categoria metodelor constructive – "distructive".

4.7.3 Metode nesecvenţiale

In categoria metodelor nesecvenţiale se pot include acele tipuri de metode mixte care, deşi se bazează pe algoritmii constructivi şi "distructivi" prezentaţi anterior, nu sunt compuse din etape independente de construire, respectiv reducere a reţelei.

❒ ***Metoda CASCOR controlată ("Controlled Growth of CASCOR")***

Metoda creşterii controlate a structurilor CASCOR (CASCORC =

CASCOR controlat) a fost propusă de Hansen, ş.a. [H29].

Algoritmul CASCOR este modificat, în sensul că, anterior introducerii fiecărui nod ascuns, acesta este supus unei faze de reducere a conexiunilor inutile. Intrucât în varianta originală CASCOR se adaugă noduri ascunse unul câte unul, se poate utiliza algoritmul de reducere OBS. Acesta implică calculul matricii hessiene complete a funcţiei cost a reţelei, motiv pentru care volumul de calcul creşte. Valorile salienţei sunt calculate ca şi în metoda OBS cu regularizare prin scăderea ponderilor, prezentată anterior. Această alegere previne divergenţa ponderilor nodurilor în cascadă spre valori infinite, care ar maximiza de altfel corelaţia.

Se consideră relaţiile valabile pentru metoda OBS cu regularizare prin scăderea ponderilor. Se presupune existenţa unui singur nod de ieşire ($N = 1$), de aceea se poate renunţa la indicele j. Pentru orice nod introdus în structură, componentele matricii hessiene a corelaţiei S sunt date de relaţia:

$$\frac{\partial^2 S}{\partial w_r \, \partial w_q} = \operatorname{sgn}\left\{\sum_{\xi=1}^{P} \left[z(\xi) - \bar{z}\right]\left[C(\xi) - \bar{C}\right]\right\}\left\{\sum_{\xi=1}^{P} \left[C(\xi) - \bar{C}\right]\frac{\partial^2 z(\xi)}{\partial w_r \, \partial w_q}\right\} \tag{4.235}$$

Semnificaţia notaţiilor este aceeaşi cu cea utilizată pentru metodele CASCOR şi OBS. Matricea ale cărei elemente se calculează cu relaţia (4.235) are rolul derivatei de ordinul doi a erorii de învăţare în metoda OBS, cu excepţia faptului că aici se doreşte maximizarea modulului corelaţiei. Valorile corespunzătoare ale salienţei sunt în acest caz date de relaţia (4.236) (cu $\boldsymbol{A} = \boldsymbol{H} - \alpha \boldsymbol{I}$, unde $\boldsymbol{I}$ este matricea unitate; indicele q al valorilor corelaţiei indică faptul că s-a eliminat ponderea w_q). S-a notat cu $\boldsymbol{u}_q$ vectorul unitate indice q.

$$\delta S_q(\mathbf{w}) = \frac{1}{2}\,\frac{\left(\mathbf{w}_0^T \mathbf{u}_q\right)^2}{\mathbf{u}_q^T \mathbf{H}^{-1} \mathbf{u}_q} - \alpha\left(\frac{\mathbf{w}_0^T \mathbf{u}_q \left(\mathbf{w}_0^T \mathbf{H}^{-1} \mathbf{u}_q\right)}{\mathbf{u}_q^T \mathbf{H}^{-1} \mathbf{u}_q} - \frac{1}{2}\,\frac{\mathbf{u}_q^T \mathbf{H}^{-2} \mathbf{u}_q}{\left(\mathbf{u}_q^T \mathbf{H}^{-1} \mathbf{u}_q\right)^2}\right) \tag{4.236}$$

Eficienţa algoritmului CASCORC este studiată de Hansen, ş.a., pentru problema "două spirale", cu scopul de a arăta că acesta conduce la o structură cu mai puţine noduri ascunse decât metoda CASCOR clasică. Performanţa metodei depinde de algoritmul de optimizare utilizat pentru nodurile ascunse. Se poate folosi algoritmul BKP, urmat de rafinarea soluţiei prin metoda Newton.

La fiecare nivel al cascadei sunt învăţate 10 noduri candidat. Cel care are valoarea corelaţiei maximă, este supus etapei de reducere a conexiunilor prin varianta OBS modificată, cu valoarea parametrului de descreştere a ponderilor

Scanned with OKEN Scanner

$\alpha = 10^{-4}$. Când valoarea amplitudinii corelaţiei scade cu 5%, se consideră încheiată faza de reducere şi nodul este instalat definitiv în reţea. Numărul nodurilor ascunse determinate se situează în intervalul 10 – 16, în timp ce varianta clasică CASCOR conduce la un număr de 11 – 17 noduri ascunse. In primul caz numărul ponderilor este 103, faţă de 146 în cel de–al doilea caz.

Structura rezultată are dimensiuni mai reduse decât cea obţinută prin aplicarea directă a algoritmului CASCOR. Utilizarea algoritmului OBS cu regularizare pentru reducerea conexiunilor inutile previne divergenţa ponderilor nodurilor în cascadă spre valori infinite [H29]. De asemenea, se evită pericolul supraînvăţării, ce apare mai ales în structurile total interconectate.

Dezavantajul metodei este necesitatea calculului matricii hessiene complete a funcţiei cost a reţelei, motiv pentru care volumul de calcul creşte.

❒ *Metoda DNAL ("Dynamic Node Architecture Learning")* (II)

Bartlett [B23] a propus o metodă de adăugare dinamică, în timpul învăţării, a nodurilor ascunse, pe baza unui criteriu din teoria informaţiei. Deşi autorul consideră metoda DNAL ("Dynamic Node Architecture Learning") de tip constructiv, aceasta implică şi paşi de reducere pe parcursul învăţării, fiind deci o metodă mixtă.

Structura neurală multistrat are M noduri de intrare, N noduri de ieşire, L straturi active (fără stratul de intrare). Setul de date de învăţare conţine P patternuri. Funcţia cost $C(\mathbf{w})$ ce trebuie minimizată este radicalul din eroarea pătratică medie, normată la N şi P.

Funcţia de activare a fiecărui nod este:

$$f(net) = \frac{1}{\pi} arctg(net) + \frac{1}{2} \qquad (4.237)$$

Structura DNAL se bazează pe un algoritm stohastic de învăţare, numit de Bartlett SOSLA ("Self – Optimizing Stochastic Learning Algorithm") (tab. 4.32). Structura iniţială a reţelei conţine numai câteva noduri ascunse. Reţeaua învăţă, până când atinge un aşa – numit platou (eroarea nu mai descreşte semnificativ în timp). Se adaugă un nod ascuns, având ponderile iniţiale aleatoare. Pe măsură ce reţeaua continuă să înveţe, ponderile sale se modifică, astfel încât nodul ascuns introdus, să influenţeze din ce în ce mai mult valorile de ieşire ale structurii. Când eroarea nu mai descreşte, înseamnă că s-a atins din nou un platou, trebuie adăugat un nou nod ascuns, etc. Procesul este reluat până când eroarea la ieşirea RPM scade sub un prag stabilit.

In oricare dintre platourile atinse, se pot elimina din reţea noduri care au importanţă redusă. La sfârşitul învăţării, se pot elimina de asemenea, noduri,

Scanned with OKEN Scanner

pentru a încerca să se obţină rezolvarea problemei cu o structură de dimensiuni mai mici. Dacă însă, eroarea creşte semnificativ prin eliminarea nodurilor, acestea sunt reintroduse în reţea. Se consideră reţea "satisfăcătoare" cea care are dimensiunea cea mai mică şi obţine performanţele dorite.

Problemele studiate cu ajutorul algoritmului DNAL sunt: problema SAU – EXCLUSIV NEGAT (XNOR), problema decodării binare 1 la 8 şi predicţia dinamicii populaţiei animale. Rezultatele au fost comparate cu cele obţinute prin aplicarea relaţiilor empirice Hecht – Nielsen şi Upadhyaya – Eryrek. Pentru cazul problemei XNOR, rezultatele comparative sunt prezentate în tab. 4.33. Structura iniţială a fost 3–1–8. Se observă că, pe măsură ce eroarea descreşte, structura reţelei creşte şi descreşte repetat, ajungând în final să aibă dimensiuni mai mici decât cele indicate de relaţiile empirice.

Metoda are avantajul numărului redus al nodurilor structurii rezultate. Conduce într-un interval de timp rezonabil, la o structură de dimensiuni mici, utilizabilă, care are în majoritatea cazurilor, performanţe mai bune decât reţelele dimensionate prin relaţii empirice.

Un dezavantaj este faptul că, metoda nu garantează obţinerea unei reţele cu performanţe mai bune din punct de vedere al generalizării, decât ale altor structuri.

Tab. 4.32 Algoritmul SOSLA pentru structura cu DNAL

❶ Se aleg valori aleatoare pentru ponderi şi se evaluează valoarea funcţiei cost la momentul $k = 0$, $C(\mathbf{w}; 0)$.
❷ Fiecare pondere este modificată şi se evaluează din nou valoarea funcţiei cost, obţinându-se valoarea $C(\mathbf{w}; k + 1)$.
Dacă $C(\mathbf{w}; k + 1) \geq C(\mathbf{w}; k)$, atunci $k \leftarrow k + 1$ şi se reia pasul ❷, altfel continuă cu pasul ❸.
❸ Se stochează cel mai bun set de ponderi şi se renunţă la setul de ponderi pentru care valoarea funcţiei cost este maximă.
❹ Se modifică criteriul de selecţie a parametrilor, pe baza informaţiei obţinute în pasul ❷.
❺ Dacă reţeaua învaţă încet, se adaugă un nod în stratul ascuns.
❻ Dacă valoarea totală a funcţiei cost este acceptabilă, $C(\mathbf{w}; k) < \varepsilon$, sau dacă un nod ascuns are importanţă redusă ori este redundant, atunci se elimină nodul din reţea.
❼ Dacă structura reţelei oscilează în jurul unei structuri anumite, STOP; altfel, salt la pasul ❷.

Scanned with OKEN Scanner

Tab. 4.33 Rezultatele metodei DNAL pentru problema SAU – EXCLUSIV NEGAT şi comparaţie cu rezultatele empirice

DNAL			Metode empirice					
			Hecht – Nielsen			Upadhyaya–Eryurek		
M	H	N	M	H	N	M	H	N
3	1	8						
3	1	8						
3	2	8						
3	2	8						
3	3	8						
3	3	8						
3	4	8						
3	4	8						
3	3	8						
3	3	8						
3	4	8						
3	4	8						
3	3	8						
3	3	8	3	5	8	3	4	8

4.8 COMPARAŢIE INTRE METODELE DE PROIECTARE A STRUCTURILOR NEURALE "FEEDFORWARD"

Comparaţia metodelor de proiectare a structurilor "feedforward" prezentate (bazate pe criterii statistice, metode de tip constructiv, "distructiv", metode mixte), este dificilă, mai ales datorită aplicării acestora pentru rezolvarea unor probleme de test ("benchmark") diferite.

In tabelul 4.34, propunem o comparaţie calitativă, ce include observaţii generale referitoare la structura generată, capacitatea de generalizare şi timpul consumat pentru proiectare.

S-au marcat cu semnele: 👍 şi 👎 rezultatele bune (avantaje), respectiv rezultatele mai slabe (dezavantaje).

Scanned with OKEN Scanner

Tab. 4.34 Comparaţie generală între diferite tipuri de metode de proiectare a structurilor neurale "feedforward"

Metode	Structura generată	Generalizare	Timp consumat	Comparaţie cu BKP
Bazate pe criterii statistice	Depinde de metodă *Nu* optimală 👎	Depinde de metodă. In general bună 👍	In general mediu 👍	👍
Constructive	Depinde de metodă *Nu* optimală 👎	Depinde de metodă. In general bună 👍	Depinde de metodă. In general mediu 👍	👍
"Distructive"	Mică 👍 *Nu* optimală 👎	Depinde de metodă. In general bună 👍	Depinde de metodă. Mare 👎, pentru metodele cu reînvăţare, mediu 👍 altfel	👍
Mixte	Depinde de metodă. In general mică sau medie 👍 *Nu* optimală 👎	Depinde de metodă. In general bună 👍	Depinde de metodă. In general mediu 👍 sau mare 👎	👍

Scanned with OKEN Scanner

4.9 REZUMATUL CAPITOLULUI

Scopul acestui capitol a fost precizarea principiilor generale de selecţie a modelului (parsimonie, coerenţă şi admisibilitate a datelor, stabilitate structurală, "excelenţă", dimensionalitate şi interpretabilitate), care stau şi la baza proiectării RNA, precum şi prezentarea unei sinteze a metodelor de proiectare a structurilor neurale "feedforward". După clasificarea generală a acestora, ne-am referit la metode: empirice, bazate pe criterii statistice, ontogenice constructive, ontogenice "distructive" şi ontogenice mixte. Prezentarea fiecăruia dintre aceste tipuri a constat în precizarea problemei de proiectare, clasificarea metodelor din cadrul tipului respectiv, prezentarea acestora cu avantaje şi dezavantaje. In finalul fiecărei secţiuni au fost incluse tabele centralizatoare ale metodelor, cu comparaţii calitative referitoare la dimensiunile structurilor obţinute, capacitatea de generalizare, timpul consumat şi problema de studiu.

Capitolul conţine, de asemenea, o metodă mixtă proprie, de tip constructiv–"distructiv", aplicată pentru proiectarea filtrelor neurale. Etapa constructivă se bazează pe algoritmul CASCOR, cu structură iniţială modificată, iar cea "distructivă" foloseşte algoritmul OBD. Au fost deduse relaţiile matematice necesare calculului senzitivităţilor conexiunilor structurii neurale, obţinută prin aplicarea algoritmului CASCOR, precum şi cele referitoare la evaluarea numărului parametrilor reţelei obţinute şi la numărul operaţiilor de "înmulţire + adunare" efectuate. Metoda mixtă secvenţială propusă a fost aplicată pentru refacerea imaginii Lenna, de dimensiune 255 x 255 pixeli. Rezultatele obţinute au fost comparate cu cele ale structurii fixe care învaţă cu algoritmul BKP, precum şi cu cele obţinute de filtrul Wiener.

Scanned with OKEN Scanner

5 PROIECTAREA REȚELELOR NEURALE "FEEDFORWARD" CU CONSTRÂNGERI DE SIMETRIE

5.1 NECESITATEA IMPUNERII CONSTRÂNGERILOR DE SIMETRIE ÎN PROIECTAREA REȚELELOR NEURALE "FEEDFORWARD"

Definiția 5.1 Simetria este o proprietate a unui ansamblu spațial, de a fi alcătuit din elemente reciproc corespondente şi de a prezenta, pe această bază, anumite regularități; reprezintă corespondența exactă (ca formă, poziție, etc.) între părțile opuse ale unui tot [DEX].

Impunerea constrângerilor de simetrie în proiectarea rețelelor neurale "feedforward" poate fi abordată din mai multe puncte de vedere. Constrângerile se pot referi la setul datelor de învățare, la ecuațiile cu derivate parțiale din algoritmul de învățare, sau la topologia neurală [C46]. In continuare vom avea în vedere ultima dintre aceste variante, adică simetria structurală.

Sinteza prezentată în cap. 4 indică existența unui număr mare de metode de proiectare, al căror scop este optimizarea arhitecturii neurale, în sensul obținerii unei structuri compacte, de mici dimensiuni, cu performanțe de generalizare acceptabile. Dacă se utilizează metode "distructive", atunci rețelele neurale obținute sunt în general simple, parțial interconectate [M81], dar nesimetrice. Lipsa simetriei structurii neurale impune proiectantului limite de realizare a implementării hardware, precum şi a simulărilor software. In cazul structurilor simetrice, stocarea parametrilor (ponderi, praguri) necesită spațiu redus, iar programele pentru procesoare digitale de semnal se simplifică. Intrucât deseori, în teoria circuitelor, se impun condiții de simetrie şi reciprocitate circuitelor, rezultă că, şi în cazul structurilor neurale, problema proiectării rețelelor neurale simetrice este justificată.

Pe de altă parte, se poate face deosebirea între constrângeri de tip "hard" şi "soft". Primele nu trebuie încălcate, deoarece costul unei astfel de acțiuni tinde la infinit. Nerespectarea ultimelor are un cost finit [W31]. Din această perspectivă, putem considera că, constrângerile de simetrie sunt de tip "soft".

Printre puținele studii care au drept scop simetrizarea RNA, se numără cel realizat de Shawe – Taylor [T23], care a urmărit obținerea unor ponderi cu valori

Scanned with OKEN Scanner

identice[21], astfel încât ieşirea reţelei să fie invariantă pentru un set de transformări efectuate asupra datelor de intrare. Yang, Yin, Gabbouj, ş.a. [Y12] au studiat simetrizarea topologiei filtrelor, impunând conservarea unor detalii ale semnalului de intrare. Ponderile pot avea valori diferite, dar poziţia conexiunilor respective trebuie să fie simetrică în structură.

Impunerea constrângerilor de simetrie structurală în etapa de învăţare este o idee riscantă. Existenţa unor simetrii interne ale structurii "feedforward" conduce la apariţia mai multor minime ale funcţiei cost, periodicităţi ale acesteia, sau chiar zone aproape plate [H19]. Minimele multiple se datorează posibilităţii permutării nodurilor în cadrul unui strat (pentru H noduri ascunse, se obţin $H!$ situaţii echivalente), sau schimbării semnului tuturor ponderilor conexiunilor care intră şi care ies într-un / dintr-un nod ascuns (pentru H noduri ascunse cu funcţie de activare simetrică faţă de zero, se obţin 2^H situaţii echivalente) [T24].

Dacă etapa de învăţare se desfăşoară fără simetrizarea structurii, atunci există posibilitatea utilizării unor metode "distructive", al căror rezultat să fie structuri simetrice cu dimensiuni minime. Aceasta este abordarea propusă în continuare, subliniind faptul că, poziţia conexiunilor trebuie să fie simetrică în reţea, fără a impune valori identice ale ponderilor acestora. Vom urmări, de asemenea, reducerea structurii neurale până la obţinerea unor matrici de ponderi rare.

5.2 FORMULAREA PROBLEMEI DE PROIECTARE NEURALĂ CU CONSTRÂNGERI DE SIMETRIE

Dacă se impun constrângeri de simetrie, problema de proiectare (4.1) devine:

Minimizează dimensiunile reţelei neurale
cu constrângerile că,
aceasta este capabilă să implementeze setul datelor de învăţare (5.1)
şi
conexiunile dintre noduri au poziţii simetrice în structura finală
şi
ponderile conexiunilor simetrice nu sunt neapărat egale

[21] Constrângerea referitoare la valorile simetrice ale ponderilor corespunde celei de-a treia legi Newton din mecanică [W31]

Scanned with OKEN Scanner

Metodele propuse pentru rezolvarea problemei de proiectare cu constrângeri de simetrie, se bazează pe modificarea algoritmului OBD prezentat în cap. 4.6 (MOBD şi SVD–MOBD), precum şi pe efectuarea unei transformări de reflexie asupra matricii ponderilor (TOBD).

5.3 METODA MOBD ("MODIFIED OPTIMAL BRAIN DAMAGE")

In lucrarea [D207] a fost propus un algoritm "distructiv" de proiectare a RNA "feedforward" cu constrângeri de simetrie, numit MOBD ("Modified Optimal Brain Damage"). La fiecare pas, este eliminată atât ponderea conexiunii având salienţa minimă, cât şi "perechea" sa simetrică din punct de vedere al poziţiei în reţea.

In RNA cu M noduri de intrare, H noduri ascunse şi N noduri de ieşire, dacă salienţa minimă, calculată conform algoritmului OBD, se obţine pentru o conexiune dintre stratul de intrare şi cel ascuns (cu ponderea w_{ih}), atunci ponderile corespunzătoare sunt "şterse" conform relaţiei (5.2). Dacă salienţa minimă se obţine pentru o conexiune dintre stratul ascuns şi cel de ieşire (cu ponderea v_{hj}), atunci ponderile corespunzătoare sunt eliminate conform relaţiei (5.3). Parantezele drepte marchează operatorul "parte întreagă", iar $1 \le i \le M$, $1 \le h \le H$, $1 \le j \le N$.

$$\text{Elimină} \begin{cases} w_{ih} \text{ şi } w_{M-i+1,h} & \text{dacă } i = \left[\frac{M}{2}\right] \text{ şi M par} \\ w_{ih} \text{ şi } w_{M-i,h} & \text{dacă } i = \left[\frac{M}{2}\right] \text{ şi M impar} \\ w_{ih} & \text{dacă } i = \left[\frac{M}{2}\right] + 1, \ \forall M \end{cases} \tag{5.2}$$

$$\text{Elimină} \begin{cases} v_{hj} \text{ şi } v_{H-h+1,j} & \text{dacă } h = \left[\frac{H}{2}\right] \text{ şi H par} \\ v_{hj} \text{ şi } v_{H-h,j} & \text{dacă } h = \left[\frac{H}{2}\right] \text{ şi H impar} \\ v_{hj} & \text{dacă } h = \left[\frac{H}{2}\right] + 1, \ \forall H \end{cases} \tag{5.3}$$

Algoritmul MOBD, prezentat în tab. 5.1, simetrizează structura neurală. Astfel, pentru matricea ponderilor conexiunilor dintre stratul de intrare şi stratul

ascuns, de dimensiuni H x M, simetria se realizează faţă de axa constituită din elementele $w_{\frac{M+1}{2},H}$.

Eliminarea unei singure conexiuni, conform metodei OBD standard, este justificată de aproximaţia iniţială referitoare la diagonalitatea matricii hessiene. Eroarea introdusă în funcţia cost poate fi aproximată prin valoarea salienţei conexiunii eliminate.

Dacă salienţa , calculată cu relaţia (4.144), este minimă pentru conexiunea $w_{i^*h^*}$, unde $1 \le i^* \le M$, $1 \le h^* \le H$, atunci anularea a două ponderi conform metodei MOBD, introduce o eroare dată de relaţia (5.4), cazul cel mai defavorabil fiind acela în care perechea sa simetrică are salienţă maximă.

$$e = s_{i^*h^*} + s_{\text{pereche simetrica}} \le s_{i^*h^*} + \max_{1\le i\le M}\left\{\max_{1\le h\le H}\left(s_{ih}\right)\right\}, \tag{5.4}$$

Dacă salienţa, calculată cu relaţia (4.142) este minimă pentru conexiunea $v_{h^*j^*}$, unde $1 \le h^* \le H$, $1 \le j^* \le N$, atunci anularea a două ponderi conform metodei MOBD, introduce o eroare dată de relaţia (5.5), cazul cel mai defavorabil fiind, de asemenea, acela în care perechea sa simetrică are salienţă maximă.

$$e = s_{h^*j^*} + s_{\text{pereche simetrica}} \le s_{h^*j^*} + \max_{1\le i\le M}\left\{\max_{1\le h\le H}\left(s_{ih}\right)\right\} \tag{5.5}$$

Imbunătăţirea performanţelor algoritmului MOBD se poate realiza prin renunţarea la aproximarea diagonalităţii matricii hessiene, dar aceasta presupune rescrierea algoritmului luând în considerarea întreaga matrice $\boldsymbol{H}$, ceea ce implică o creştere semnificativă a timpului de calcul şi spaţiului de memorie necesar.

Tab. 5.1 Algoritmul MOBD

① Se învaţă reţeaua, până când se obţine eroarea de învăţare dorită. ② Se calculează valorile salienţelor, conform algoritmului OBD. ③ Se elimină ponderea având salienţa minimă, împreună cu "perechea" sa simetrică. Se utilizează relaţia (5.2), dacă acestea se referă la conexiunile dintre stratul de intrare şi cel ascuns, respectiv relaţia (5.3) dacă se referă la conexiunile dintre stratul ascuns şi cel de ieşire. ④ Se învaţă din nou reţeaua. ⑤ Se repetă paşii ② – ④, până când structura neurală se micşorează suficient, iar eroarea de test nu scade sub un prag fixat.

Scanned with OKEN Scanner

5.4 METODA SVD–MOBD ("MODIFIED OPTIMAL BRAIN DAMAGE WITH SINGULAR VALUE DECOMPOSITION")

5.4.1 Utilizarea transformărilor pentru proiectarea RNA

In studiul formal al modelelor neurale, deseori sunt utilizate metode clasice. Andreas Weigend sublinia că, de exemplu, "este importantă extinderea aplicării tehnicilor de analiză liniară cât este posibil, dar nu mai departe: aşa cum este o prostie ignorarea rezultatelor obţinute de analiza liniară, tot astfel este periculoasă aplicarea lor fără discernământ în cazul neliniar. Un exemplu superb de aplicare utilă a tehnicilor liniare în cazul neliniar este analiza metodei gradientului negativ utilizând valori proprii şi vectori proprii ai suprafeţei de eroare". Pornind de la această idee, este semnificativ faptul că, în proiectarea RNA s-au utilizat analiza componentelor principale (PCA, "Principal Component Analysis"), transformarea Karhunen – Loéve (ambele bazate pe calculul valorilor proprii ale seturilor de date), transformări ortogonale precum SVD ("Singular Value Decomposition"), QR, etc. Cele două abordări au constat în: ① utilizarea acestora ca etape anterioare învăţării sau etape intermediare în timpul învăţării; ② utilizarea unor legi de învăţare care să realizeze selecţia componentelor principale (Sanger, Oja, etc.)

Utilizarea componentelor principale realizează de fapt proiectarea datelor din spaţiul de intrare într-un spaţiu H – dimensional, în care dispersia semnalului pe cele H direcţii este maximă [W31]. Aplicaţii ale PCA pentru proiectarea RNA menţionează Haykin [H56], au propus Chen şi Chang [C47], Baldi şi Hornik [B42], Park, Murray şi Chen [P30]. Cei din urmă utilizat PCA pentru determinarea structurii reţelei "feedforward" perceptron multistrat. Algoritmul propus în lucrarea [P30] presupune învăţarea reţelei cu un număr suficient de mare de noduri ascunse. Se calculează matricea de covariaţie de dimensiuni H x H, a valorilor de ieşire din stratul ascuns şi se determină valorile proprii ale acesteia. Dacă numărul valorilor proprii mai mari decât unitatea, este mai mic decât H, atunci se păstrează un număr de noduri ascunse egal cu minimul dintre cele două valori. Altfel, nodurile ascunse nu sunt redundante.

Metoda este simplă şi nu face nici o ipoteză asupra distribuţiei datelor de intrare. Dezavantajul major este însă, dificultatea calculului matricii de covariaţie pentru seturi de date de intrare de mari dimensiuni, precum şi dificultatea interpretării componentelor principale extrase.

O soluţie pentru prima observaţie este utilizarea tehnicii SVD. Kanjilal şi Benrjee [K49] au propus aplicarea SVD, precum şi a factorizării QR pentru reducerea structurii neurale şi pentru selecţia setului optim de date de învăţare.

Scanned with OKEN Scanner

Matricea valorilor de ieşire din stratul ascuns, pentru toate patternurile de intrare, este descompusă SVD. Numărul valorilor singulare dominante determinate indică numărul nodurilor ascunse care trebuie păstrate. Alegerea numărului valorilor singulare care sunt semnificative se face euristic. O soluţie similară este cea utilizată de Psichogios şi Ungar [P29], însă eliminarea nodurilor redundante are loc în timpul învăţării, a cărei durată creşte considerabil.

Wilson şi Tufts [W32] au calculat matricea de autocorelaţie a ieşirilor nodurilor ascunse, pe care au descompus-o aplicând metoda SVD.

Trebuie remarcat faptul că, utilizarea tehnicilor liniare pentru proiectarea RNA nu este o idee nouă. Deşi PCA se aplică de obicei reţelelor liniare [B42], Kung preciza faptul că, din punct de vedere teoretic, se poate arăta că, neliniaritatea nodurilor ascunse nu oferă un avantaj suplimentar [K27]. Pe de altă parte, valorile singulare pot fi calculate mai eficient, cu precizie mai mare decât valorile proprii, iar algoritmii sunt stabili din punct de vedere numeric. De aceea, metodele bazate pe SVD sunt superioare celor bazate pe calculul valorilor proprii [K49].

5.4.2 Descompunerea SVD

Pentru o matrice arbitrară $\boldsymbol{M}$, cu dimensiunile m x n şi rangul r, există numerele $\lambda_1 \geq \ldots \lambda_r > 0$, numite valori singulare ale $\boldsymbol{M}$, o matrice ortogonală $\boldsymbol{L}$, de dimensiuni m x m şi o matrice ortogonală $\boldsymbol{E}$, de dimensiuni n x n, astfel încât matricea [C16, K49]:

$$\mathbf{B} = \mathbf{L}^T \mathbf{M} \mathbf{E} \tag{5.6}$$

să fie diagonală, cu dimensiunile m x n, având forma:

$$\mathbf{B} = \begin{pmatrix} \Lambda & 0 \\ 0 & 0 \end{pmatrix} \tag{5.7}$$

unde $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_r)$ este matrice diagonală şi $r = \min(m, n)$. Descompunerea (5.8) se numeşte descompunerea SVD a matricii $\boldsymbol{M}$ (descompunerea nu este neapărat unică).

$$\mathbf{M} = \mathbf{L} \mathbf{B} \mathbf{E}^T \tag{5.8}$$

Coloanele matricii $\boldsymbol{E}^T$ au norma unitară şi sunt vectorii proprii reciproc perpendiculari (vectori singulari drepţi) ai $\boldsymbol{M}^T \boldsymbol{M}$, iar $\lambda_1^2, \cdots, \lambda_r^2$ sunt valorile proprii nenule ale $\boldsymbol{M}^T \boldsymbol{M}$. Similar, coloanele matricii $\boldsymbol{L}$ au norma unitară şi sunt

Scanned with OKEN Scanner

vectorii proprii reciproc perpendiculari (vectori singulari stângi) ai $\boldsymbol{M}\ \boldsymbol{M}^T$. Elementele matricii $\boldsymbol{B}$ se numesc valori singulare ale matricii $\boldsymbol{M}$ [B42, K49].

Energia seriei reprezentate de elementele M_{ij} ale matricii $\boldsymbol{M}$ este dată de norma Frobenius [D39]:

$$\|\mathbf{M}\|_F^2 = \mathrm{tr}\left\{\mathbf{M}\,\mathbf{M}^T\right\} = \sum_{i=1}^{m}\sum_{j=1}^{n} M_{ij}^2 = \sum_{i=1}^{\min(m,\,n)} \lambda_i^2 \qquad (5.9)$$

unde $tr\{\bullet\}$ este operatorul "urma matricii".

Dacă un număr q din cele r valori singulare, sunt dominante, se poate selecta un subset m x q al matricii $\boldsymbol{M}$ ($q < n$), care conţine cea mai mare parte a informaţiei din matricea $\boldsymbol{M}$.

5.4.3 Utilizarea descompunerii SVD în metoda MOBD

Ideile care stau la baza metodei SVD–MOBD sunt:

❶ Examinarea corelaţiei valorilor de ieşire din nodurile ascunse, după etapa de învăţare, poate indica existenţa unor noduri redundante sau nu. Dacă vectorii de ponderi pentru două noduri ascunse sunt ortogonali, atunci coeficientul de corelaţie este nul, altfel este nenul. Existenţa unui număr nenul de valori singulare mici în matricea $\boldsymbol{B}$, rezultată din descompunerea SVD a matricii ieşirilor din nodurile ascunse, indică existenţa unor noduri ascunse redundante în reţea, care se pot elimina. Există însă cazuri în care nu se poate lua o decizie de eliminare, valorile singulare fiind apropiate.

❷ Intrucât prin învăţare, se urmăreşte obţinerea unor vectori de ponderi ortogonali, este justificată utilizarea unor transformări asupra matricilor de ponderi, care să realizeze acest obiectiv.

❸ Oh şi Lee [O10] au arătat că, transformările neliniare de tip sigmoidal reduc corelaţia dintre ieşirile nodurilor ascunse. Putem presupune că, transformarea neliniară sigmoidală aplicată direct matricii de ponderi dintre stratul de intrare şi cel ascuns, reduce corelaţia dintre vectorii de ponderi.

❹ Pre– sau post– multiplicarea cu o matrice ortonormală nu afectează norma Frobenius [D39]. In consecinţă, norma Frobenius a matricii $\boldsymbol{M}$ poate fi aproximată cu o eroare cunoscută, de norma matricii $\boldsymbol{B}$ din descompunerea (5.8). Dacă matricea $\boldsymbol{M}$ este o matrice de ponderi (a conexiunilor dintre stratul de intrare şi cel ascuns, sau dintre stratul ascuns şi cel de ieşire), norma sa Frobenius poate fi aproximată prin norma matricii diagonale rezultate din descompunerea SVD, în care s-au păstrat valorile singulare dominante.

Algoritmul SVD–MOBD propus este prezentat în tab. 5.2

Scanned with OKEN Scanner

Tab. 5.2 Algoritmul SVD–MOBD

① Se învaţă reţeaua, până când se obţine valoarea dorită a erorii de învăţare.
② Se calculează matricea $\boldsymbol{Z}$ a valorilor de ieşire din stratul ascuns:

$$\mathbf{Z} = f(\mathbf{W}\ \mathbf{X}) \qquad (5.10)$$

unde $\mathbf{W}$ este matricea ponderilor conexiunilor dintre stratul de intrare şi stratul ascuns, iar $\boldsymbol{X}$ este matricea datelor de intrare
③ Se descompune SVD matricea $\boldsymbol{Z}$:

$$\mathbf{Z} = \mathbf{L}_1\ \mathbf{B}_1\ \mathbf{E}_1^T \qquad (5.11)$$

Dacă există valori singulare cu cel puţin un ordin de mărime mai mici decât valoarea singulară maximă, nodurile ascunse corespunzătoare se elimină din reţea. Salt la pasul ⑥.
Altfel, continuă cu pasul ④.
④ Se descompune SVD matricea $\boldsymbol{V}$ a ponderilor conexiunilor dintre stratul ascuns şi cel de ieşire:

$$\mathbf{V} = \mathbf{L}_2\ \mathbf{B}_2\ \mathbf{E}_2^T \qquad (5.12)$$

Dacă există valori singulare cu cel puţin un ordin de mărime mai mici decât valoarea singulară maximă, nodurile ascunse corespunzătoare se elimină din reţea.
⑤ Dacă eroarea de test este mai mare decât un prag prestabilit ε_1, se reintroduc nodurile ascunse eliminate şi se continuă cu pasul ⑥.
⑥ Se utilizează metoda MOBD (tab. 5.1) pentru reducerea conexiunilor dintre stratul de intrare şi cel ascuns.

5.5 METODA TOBD ("TRIDIAGONAL OPTIMAL BRAIN DAMAGE")

5.5.1 Matrici rare

Propoziţia 5.1 O matrice pătrată de dimensiuni m x m este rară dacă, conţine un număr mic τ de elemente nenule, unde $\tau << m^2$.

Scanned with OKEN Scanner

Definiția 5.2 O matrice pătrată de dimensiuni m x m este rară dacă, atunci când m crește, numărul elementelor nenule τ, este $\tau < m^{1+c}$, $0 \le c \le 1$ [R200].

Extinzând propoziția 5.1, se poate spune că:

Propoziția 5.2 O matrice de dimensiuni m x n, este rară dacă $\tau << m\,n$.

Prin analogie cu definiția 5.2, se poate formula definiția:

Definiția 5.3 O matrice de dimensiuni m x n este rară dacă, atunci când m și n cresc, numărul elementelor nenule τ, este $\tau < \min(m, n)^{1+c}$, $0 \le c \le 1$.

Matricile rare sunt caracterizate cantitativ de densitatea matricii, definită ca raport între numărul elementelor nenule și cel al elementelor nule: $\tau / (m\,n - \tau)$. Valorile uzuale se află în intervalul 0.15 % ....3 % [R200].

Există matrici rare speciale, între care se numără matricile bandă, matricile tridiagonale (caz particular al formei diagonale de tip bandă), matricile triunghiulare, matricile "aproape triunghiulare" (matrici Hessenberg), matricile rare simetrice, etc.

Definiția 5.4 O matrice $\boldsymbol{M}$ este matrice rară de tip bandă, cu lățimea benzii B, dacă elementele M_{ij} sunt [S209]:

$$\begin{cases} 0 & \text{pentru } |i - j| \ge B \\ M_{ij} & \text{altfel} \end{cases} \qquad (5.13)$$

Dacă toate elementele sunt nule pentru $i \ne j$, matricea se numește pseudo – diagonală (dacă nu este pătrată) și diagonală (dacă este matrice pătrată) [D39].

Definiția 5.5 Dacă $B = 2$ în definiția 5.4, matricea $\boldsymbol{M}$ este tridiagonală.

O matrice poate fi redusă la oricare dintre formele speciale într-un număr finit de rotații plane (metoda Givens) sau reflexii (metoda Householder) [S210].

Scanned with OKEN Scanner

5.5.2 Argumente în favoarea obţinerii matricilor tridiagonale de ponderi pentru structuri neurale simetrice

Există mai multe argumente în favoarea obţinerii matricilor tridiagonale de ponderi:

❶ Structura neurală caracterizată de o matrice de ponderi tridiagonală este simetrică.

❷ Metodele de stocare a valorilor elementelor matricilor tridiagonale nu utilizează câmpuri secundare pentru găsirea elementelor matricii [R200], deci timpul de căutare este redus, chiar dacă se stochează uneori şi o parte a elementelor nule.

❸ Implementările hardware ale structurii neurale caracterizată de matrici de ponderi tridiagonale se realizează mai uşor şi sunt mai ieftine decât cele ale reţelelor total interconectate, datorită spaţiului redus de stocare al parametrilor şi vitezei ridicate de realizare a operaţiilor matematice (multiplicări şi adunări în structurile neurale), faţă de cazul structurilor total interconectate.

Stocarea unei matrici oarecare de dimensiuni m x n relativ scăzute, necesită alocarea unei zone bidimensionale, în care se păstrează atât elementele nule, cât şi cele nenule. Dacă matricea conţine un număr mare de elemente nule, când m şi n cresc, metoda de stocare standard devine ineficientă. In acest caz, numai elementele nenule trebuie păstrate, economisind astfel atât memorie / spaţiu pe chipul de siliciu, cât şi timp. Intrucât zona de stocare nu mai coincide în acest caz cu cea utilizată pentru matricea iniţială, sunt necesare tehnici de identificare a poziţiilor elementelor matricii.

Pentru o matrice $\boldsymbol{M} = \{M_{ij}\}$ diagonală de tip bandă de ordin n, cu lăţimea benzii B, elementul M_{ij} cu $j < i$, este stocat la adresa [R200]:

$$\begin{cases} j + \dfrac{i\,(i-1)}{2} & \text{pentru} \quad i \le B \\ (B-1)\left(i - \dfrac{B}{2}\right) + j & \text{pentru} \quad i > B \end{cases} \tag{5.14}$$

Elementul diagonal M_{ii} este stocat la adresa:

$$\begin{cases} \dfrac{i\,(i+1)}{2} & \text{pentru} \quad i \le B \\ \dfrac{B\,(2\,i - B + 1)}{2} & \text{pentru} \quad i > B \end{cases} \tag{5.15}$$

Pentru extragerea elementelor liniei i, este suficientă determinarea adresei

elementului diagonal $M_{i-1,i-1}$.

Pentru o matrice tridiagonală $B = 2$, deci din relaţia (5.14) rezultă că, elementul M_{ij} cu $j < i$, este stocat la adresa:

$$\begin{cases} j + \dfrac{i(i-1)}{2} & \text{pentru } i \le 2 \\ j + i - 1 & \text{pentru } i > 2 \end{cases} \tag{5.16}$$

Din (5.15) rezultă că, elementul diagonal M_{ii} este stocat la adresa:

$$\begin{cases} \dfrac{i(i+1)}{2} & \text{pentru } i \le 2 \\ 2i - 1 & \text{pentru } i > 2 \end{cases} \tag{5.17}$$

❹ Simularea reţelelor neurale pe procesoare de semnal necesită stocarea coeficienţilor (ponderi, praguri). Existenţa unor valori simetrice implică o procesare mai uşoară şi mai rapidă, datorită numărului mic de operaţii de "înmulţire + adunare".

5.5.3 Transformarea de reflexie

Matricile $\boldsymbol{M}$ şi $\mathcal{J}\boldsymbol{M}\mathcal{J}^{-1}$, cu $\mathcal{J}$ matrice nesingulară oarecare, au aceleaşi valori proprii. Matricile $\boldsymbol{M}$ şi $\mathcal{J}\boldsymbol{M}\mathcal{J}^{-1}$ se numesc similare, iar transformarea $\mathcal{J}\boldsymbol{M}\mathcal{J}^{-1}$ a matricii $\boldsymbol{M}$ (de dimensiuni m x m) se numeşte transformare de similaritate. Pentru cazul reflexiilor, care sunt transformări ortogonale (deci, păstrează norma euclidiană), matricea $\mathcal{J}(\boldsymbol{q})$ care le caracterizează este ortogonală, având forma [S210]:

$$\mathcal{J}(\mathbf{q}) = \mathbf{I} - 2\,\mathbf{q}\,\mathbf{q}^T, \text{ cu } \mathbf{q}\,\mathbf{q}^T = 1 \tag{5.18}$$

Este evident că matricea $\mathcal{J}(\boldsymbol{q})$ este ortogonală, întrucât:

$$(\mathbf{I} - 2\,\mathbf{q}\,\mathbf{q}^T)(\mathbf{I} - 2\,\mathbf{q}\,\mathbf{q}^T) = \mathbf{I} - 4\,\mathbf{q}\,\mathbf{q}^T + 4\,\mathbf{q}\,\mathbf{q}^T\mathbf{q}\,\mathbf{q}^T = \mathbf{I} \tag{5.19}$$

unde $\boldsymbol{I}$ este matricea unitate. De asemenea, $\mathcal{J}(\boldsymbol{q})$ este simetrică, deci:

$$\mathcal{J}^{-1}(\mathbf{q}) = \mathcal{J}^T(\mathbf{q}) = \mathcal{J}(\mathbf{q}) \tag{5.20}$$

Dacă matricea $\boldsymbol{M}$ se înmulţeşte la stânga cu $\mathcal{J}(\boldsymbol{q})$ rezultă:

Scanned with OKEN Scanner

$$\mathbf{M}^{(s)} = \mathfrak{I}(\mathbf{q})\,\mathbf{M} \tag{5.21}$$

atunci fiecare coloană m_j, a matricii M, cu $1 \le j \le m$, se transformă independent, în coloana $m_j^{(s)}$ a matricii $M^{(s)}$.

Similar, fiecare linie m_i a matricii M, cu $1 \le i \le m$, se transformă independent, în linia $m_i^{(d)}$ a matricii $M^{(d)}$, dacă matricea M este înmulţită cu $\mathfrak{I}(q)$ la dreapta:

$$\mathbf{M}^{(d)} = \mathbf{M}\,\mathfrak{I}(\mathbf{q}) \tag{5.22}$$

5.5.4 Utilizarea transformării de reflexie în metoda TOBD

Metoda TOBD ("Tridiagonal Optimal Brain Damage"), propusă în lucrarea [D205], se bazează pe realizarea unei transformări de reflexie, ca pas intermediar introdus în algoritmul standard OBD. Se calculează limita superioară a erorii introduse de această transformare şi se arată că structura neurală rezultată este caracterizată de o matrice de ponderi rară.

Ideile pe care se bazează metoda TOBD sunt:

❶ Matricea M din par. 5.5.3 este matricea $\mathbf{W}$ a ponderilor conexiunilor dintre stratul de intrare de dimensiune M, şi stratul ascuns de dimensiune H.

Fără a restrânge generalitatea, se presupune că: ① dimensiunile matricii $\mathbf{W}$ sunt M x H, deci coloanele sale sunt vectorii ponderilor conexiunilor care intră în fiecare nod ascuns (de exemplu, prima coloană conţine valorile ponderilor conexiunilor de la toate nodurile de intrare, la primul nod ascuns); ② $M > H$.

❷ Prin impunerea constrângerilor de simetrie, trebuie determinată matricea de transformare $\mathfrak{I}$, astfel încât transformarea de reflexie $\mathfrak{I}\,\tilde{\mathbf{W}}\,\mathfrak{I}^{-1}$, realizată asupra matricii $\tilde{\mathbf{W}}$, să aibă drept rezultat o matrice de ponderi tridiagonală. Matricea $\tilde{\mathbf{W}}$ se obţine prin completarea cu zerouri a matricii $\mathbf{W}$, până când aceasta devine matrice pătrată de dimensiuni M x M. Elementele matricii $\tilde{\mathbf{W}}$ sunt:

$$\tilde{w}_{ih} = \begin{cases} w_{ih} & \text{pentru } 1 \le i \le M,\ 1 \le h \le H \\ 0 & \text{pentru } 1 \le i \le M,\ H+1 \le h \le M \end{cases} \tag{5.23}$$

Dacă se selectează o pondere $\tilde{w}_{i^*h^*}$ care este element diagonal al matricii $\tilde{\mathbf{W}}$ (există relaţia $i^* = h^*$), atunci constrângerile de simetrie impuse sunt:

① Conexiunea respectivă se elimină ($\tilde{w}_{i^*h^*} = 0$), dar se păstrează în rețea conexiunile "vecine" caracterizate de ponderile (tridiagonale): $\tilde{w}_{h^*-1,\,h^*}$, $\tilde{w}_{h^*+1,h^*}$ ("vecini" pe aceeaşi coloană) şi $\tilde{w}_{h^*,h^*-1}$, $\tilde{w}_{h^*,h^*+1}$ ("vecini" pe aceeaşi linie). Toate celelalte conexiuni din vectorul $\tilde{\mathbf{w}}_{h^*}$ se elimină.

De exemplu, pentru o matrice inițială $\mathbf{W}$ de dimensiuni 7 x 3, matricea $\tilde{\mathbf{W}}$ are dimensiunile 7 x 7. Presupunând că ponderea selectată este w_{33}, se poate observa că vecinii săi (marcați prin săgeți) aparțin tridiagonalei matricii.

$$M = 7 \qquad \begin{array}{ccc|cccc} \multicolumn{3}{c}{H = 3} & \multicolumn{4}{c}{M - H = 4} \\ w_{11} & w_{12} & w_{13} & 0 & 0 & 0 & 0 \\ w_{21} & w_{22} & w_{23} & 0 & 0 & 0 & 0 \\ w_{31} & w_{32} & w_{33} & 0 & 0 & 0 & 0 \\ w_{41} & w_{42} & w_{43} & 0 & 0 & 0 & 0 \\ w_{51} & w_{52} & w_{53} & 0 & 0 & 0 & 0 \\ w_{61} & w_{62} & w_{63} & 0 & 0 & 0 & 0 \\ w_{71} & w_{72} & w_{73} & 0 & 0 & 0 & 0 \end{array} \tag{5.24}$$

② $\mathbf{q}\,\mathbf{q}^T = 1$ (5.25)

Dacă matricea $\tilde{\mathbf{W}}$ se înmulțește la stânga cu $\mathfrak{I}(q)$ rezultă:

$$\tilde{\mathbf{W}}^{(s)} = \mathfrak{I}(\mathbf{q})\ \tilde{\mathbf{W}} \tag{5.26}$$

atunci coloana $\tilde{\mathbf{w}}_{h^*}$, a matricii $\tilde{\mathbf{W}}$, cu $1 \le h^* \le M$, se transformă independent, în coloana $\tilde{\mathbf{w}}_{h^*}^{(s)}$ a matricii $\tilde{\mathbf{W}}^{(s)}$, conform relației:

$$\tilde{\mathbf{w}}_{h^*}^{(s)} = (\mathbf{I} - 2\,\mathbf{q}\,\mathbf{q}^T)\ \tilde{\mathbf{w}}_{h^*} = \tilde{\mathbf{w}}_{h^*} - 2\,(\mathbf{q}^T\ \tilde{\mathbf{w}}_{h^*})\,\mathbf{q} \tag{5.27}$$

Din condiția ①, impunând păstrarea ponderilor "vecine" din aceeaşi coloană, se obțin elementele vectorului $q = [q_1, q_2, q_3 \ldots q_M]^T$ date de relația (5.28). Din condiția ② şi folosind (5.28) se obține relația (5.29).

Similar, multiplicarea la dreapta a matricii cu $\mathfrak{I}(q)^T$, anulează toate elementele din linia h^*, cu excepția "vecinilor" pe linie ($\tilde{w}_{h^*,h^*-1}$, $\tilde{w}_{h^*,h^*+1}$) ai ponderii $\tilde{w}_{i^*h^*}$

$$q_i = \begin{cases} 0 & \text{pentru } i = h^* - 1 \text{ sau } i = h^* + 1 \\ \dfrac{\tilde{W}_{ih^*}}{2\sum_{m=1}^{M} q_m \tilde{W}_{mh^*}} & \text{pentru } 1 \le i < h^* - 1 \text{ sau } i = h^* \text{ sau } h^* + 1 < i \le M \end{cases} \tag{5.28}$$

$$q_i = \begin{cases} 0 & \text{pentru } i = h^* - 1 \text{ sau } i = h^* + 1 \\ \dfrac{\tilde{W}_{ih^*}}{\sqrt{\sum_{\substack{m=1 \\ m \neq h^*-1,\ m \neq h^*+1}}^{M} \tilde{W}_{mh^*}^2}} & \text{pentru } 1 \le i < h^* - 1 \text{ sau } i = h^* \text{ sau } h^* + 1 < i \le M \end{cases} \tag{5.29}$$

Astfel, transformarea globală impusă acţionează asupra unei vecinătăţi a elementului $\tilde{W}_{h^*h^*}$ din matricea de ponderi. Modul de alegere al acestuia, precum şi rezolvarea cazului în care $i^* \neq h^*$ sunt prezentate în tab. 5.3, care conţine algoritmul OBD tridiagonal (TOBD), bazat pe efectuarea unei transformări de reflexie asupra matricii ponderilor.

Se observă că, în cazul în care salienţa minimă corespunde unei conexiuni dintre stratul ascuns şi cel de ieşire, se aplică metoda MOBD pentru păstrarea simetriei acestei matrici de ponderi.

Dacă salienţa minimă corespunde unei conexiuni dintre stratul de intrare şi cel ascuns, iar ponderea corespunzătoare este element din afara diagonalei matricii de ponderi, se realizează o reducere a conexiunii conform algoritmului OBD standard, urmată de anularea tuturor conexiunilor din coloana respectivă, în afară "vecinilor" elementului diagonal. Altfel, se aplică transformarea de reflexie.

Trebuie precizat faptul că, datorită etapei de reînvăţare ⑦, matricea de ponderi care rezultă este o matrice pseudo – diagonală. Cu toate acestea, valorile din afara celor trei diagonale sunt mult mai mici decât cele din bandă.

Eroarea datorată eliminării conexiunilor dintre stratul de intrare şi cel ascuns prin algoritmul propus, poate fi aproximată prin suma salienţelor ponderilor anulate. De exemplu, în cazul eliminării ponderilor de pe coloana h^*, prin multiplicarea la stânga a matricii ponderilor cu matricea de transformare, eroarea este dată de relaţia (5.30). Numărul maxim al acestora este $M - 2$. Acestei erori i se adaugă cea datorată eliminării pe linie a ponderilor, în număr de maximum $H - 1$, prin multiplicarea la dreapta a matricii ponderilor cu matricea de transformare.

Scanned with OKEN Scanner

Tab. 5.3 Algoritmul TOBD

① Se învaţă reţeaua, până când se obţine valoarea dorită a erorii de învăţare. Se obţin matricile de ponderi $\mathbf{W} = \{w_{ih},\ 1 \le i \le M,\ 1 \le h \le H\}$, şi $\mathbf{V} = \{v_{hj},\ 1 \le h \le H,\ 1 \le j \le N\}$, pentru conexiunile dintre stratul de intrare şi stratul ascuns, respectiv dintre stratul ascuns şi cel de ieşire.
② Se calculează valorile salienţelor, conform metodei OBD.
③ Dacă ponderea care corespunde salienţei minime aparţine unei conexiuni dintre stratul ascuns şi cel de ieşire, atunci conexiunea se elimină conform relaţiei (5.3). Salt la pasul ⑦.
Dacă ponderea care corespunde salienţei minime aparţine unei conexiuni dintre stratul de intrare şi cel ascuns, notată cu $w_{i^*h^*}$, atunci continuă cu pasul ④.
④ Dacă $i^* = h^*$ pentru ponderea $w_{i^*h^*}$, continuă cu pasul ⑤, altfel salt la pasul ⑥.
⑤ Se construieşte matricea pătrată $\tilde{\mathbf{W}}$ de dimensiuni M x M, prin completarea cu zerouri a matricii $\mathbf{W}$.
Se construieşte vectorul $\boldsymbol{q} = [q_1, q_2, q_3, \ldots, q_m]^T$ cu elementele q_i date de relaţia (5.29), unde $\tilde{w}$ sunt elementele matricii $\tilde{\mathbf{W}}$.
Se construieşte matricea de transformare $\mathcal{T}$, folosind relaţia (5.18).
Se realizează transformarea $\mathcal{T}\,\tilde{\mathbf{W}}\,\mathcal{T}^{-1}$ şi apoi se normează elementele matricii rezultate. Salt la pasul ⑦.
⑥ Dacă $i^* \neq h^*$ pentru ponderea $w_{i^*h^*}$, atunci conexiunea respectivă se elimină ($w_{i^*h^*} = 0$).
Pentru elementul diagonal cu indicele $m^* = \min(i^*, h^*)$ execută pasul ⑤.
⑦ Se învaţă din nou reţeaua.
⑧ Se repetă paşii ② – ⑦, până când matricea de ponderi devine bandă diagonală şi eroarea de test este acceptabilă.

$$e_{h^*} = \sum_{\substack{1 \le i < h^*-1,\\ i = h^*,\\ h^*+1 < i \le M}}^{M} s_{ih^*} \le (M-2) \max_{\{1 \le i \le M\} \setminus \{i = h^*-1,\, i = h^*+1\}} \left(s_{ih^*}\right) \tag{5.30}$$

$$e_{i^*} = \sum_{\substack{1 \le h < i^*-1,\\ h = i^*,\\ i^*+1 < h \le H}}^{H} s_{i^*h} \le (H-1) \max_{\{1 \le h \le H\} \setminus \{h = i^*-1,\, h = i^*+1\}} \left(s_{i^*h}\right) \tag{5.31}$$

Pentru $i^* = h^*$, eroarea totală este:

$$e = e_{i^*} + e_{h^*} \le (M + H) \max_{\{1 \le i^* \le M\}} \left\{ \max_{\{1 \le h^* \le H\}} \left(s_{i^* h^*} \right) \right\} \tag{5.32}$$

Termenul din dreapta relaţiei (5.32) este o limită superioară a erorii, atunci când se elimină ponderile conexiunilor dintre două straturi succesive (cazul cel mai defavorabil fiind acela în care, sunt eliminate prin transformare conexiunile caracterizate de primele $M + H - 3$ cele mai mari valori ale salienţei).

5.6 REZULTATE EXPERIMENTALE ALE APLICĂRII METODELOR PROPUSE, DE PROIECTARE CU CONSTRÂNGERI DE SIMETRIE

5.6.1 Problema de studiu

Din punct de vedere teoretic, problema de studiu este regresia neliniară descrisă de relaţia (4.114). Se urmăreşte rezolvarea sa cu ajutorul structurii neurale "feedforward" perceptron multistrat, utilizând expresia (3.53).

Se presupune un model al datelor caracterizat de funcţia cost aditivă dată de (3.23). Minimizarea sa se poate realiza pe baza algoritmilor amintiţi în par. 3.2.2.3, problema de optimizare fiind (3.23) sau (3.25). Având în vedere obiectivul propus, de obţinere a unei structuri neurale simetrice, aceasta etapă este urmată de rezolvarea problemei de proiectare (5.1), pentru reţeaua obţinută în pasul anterior.

5.6.1.1 Setul datelor

Semnalul "chirp", reprezentat prin 5760 de eşantioane, constituie setul de date de studiu. Setul de date se partajează în două subseturi egale, primul reprezentând datele de învăţare, iar cel de-al doilea, datele de test (fig. 5.1).

Fiecare din cele două subseturi de date, compus din 2880 de eşantioane ale semnalului "chirp", a fost citit printr-o fereastră de dimensiune 7 eşantioane, care s-a deplasat cu câte un eşantion spre dreapta. Valoarea de ieşire dorită a fost cea imediat următoare ferestrei (de exemplu, valoarea a opta pentru prima fereastră). Au rezultat două matrici de intrare, fiecare având dimensiunea 7 x 2880, reprezentând datele prezentate reţelei în etapa de învăţare, respectiv de test, precum şi un vector al valorilor de ieşire dorite, de dimensiune 1 x 2880.

Scanned with OKEN Scanner

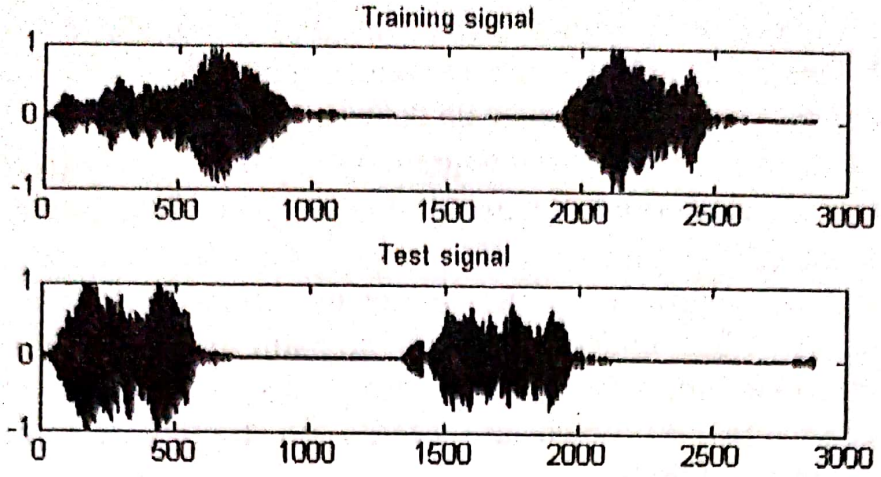


Fig. 5.1 Semnalul "chirp" de învăţare şi semnalul "chirp" de test

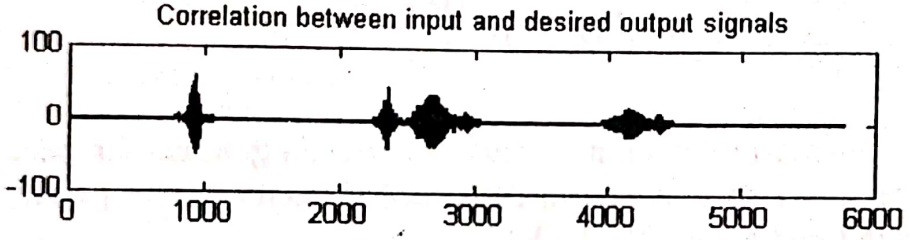


Fig. 5.2 Corelaţia dintre semnalul "chirp" de intrare şi cel de ieşire dorit

5.6.1.2 Modelul neural

❒ ***Structura reţelei neurale***

Datorită dimensiunii patternurilor de intrare şi patternurilor de ieşire dorite, rezultă că dimensiunea straturilor neurale de intrare şi de ieşire este $M = 7$, respectiv $N = 1$. Relaţia Baum – Haussler – Widrow (4.8) pentru calculul numărului de noduri ascunse, pentru $P = 2880$, $\varepsilon = 0.01$, $M = 7$, $N = 1$, conduce la inegalitatea: $H \leq 3.6$. Dacă se alege $H = 3$, atunci dimensiunile reţelei sunt 7 – 3 – 1 (7 noduri de intrare, 3 noduri ascunse şi 1 nod de ieşire).

❒ ***Algoritmul de învăţare***

Algoritmul de învăţare considerat este BKP [R3, ş.a.] cu moment.

Scanned with OKEN Scanner

La intrarea reţelei se prezintă datele din setul de învăţare construit la punctul 5.6.6.2. Funcţia de activare *f(net)* a fiecărui perceptron simplu din stratul ascuns şi de ieşire este tangenta sigmoidală dată de relaţia (3.10).

Diferenţa dintre valorile de ieşire obţinute $y_j(\xi)$, cu $1 \le j \le N$ ($N = 1$ pentru reţeaua aleasă) şi valorile dorite $d_j(\xi)$, pentru fiecare pattern de intrare $1 \le \xi \le P$, reprezintă eroarea $e_j(\xi) = d_j(\xi) - y_j(\xi)$, argument al funcţiei cost. Valoarea totală a funcţiei cost, pentru toate patternurile de intrare prezentate reţelei se calculează cu relaţia (5.33), unde normarea la numărul total al patternurilor P şi numărul perceptronilor simpli de ieşire N, face posibilă comparaţia rezultatelor obţinute de aceeaşi reţea, cu dimensiuni diferite ale stratului de ieşire, pentru seturi de date de mărimi diferite.

$$C(\mathbf{w}) = \frac{1}{N\,P}\sum_{\xi=1}^{P} C(\mathbf{w};\xi) = \frac{1}{N\,P}\sum_{\xi=1}^{P}\sum_{j=1}^{N} C\left(\mathbf{w}; e_j(\xi)\right) =$$

$$= \frac{1}{N\,P}\sum_{\xi=1}^{P}\sum_{j=1}^{N}\left\{\frac{1}{2}\left[d_j(\xi) - y_j(\xi)\right]^2\right\} \qquad (5.33)$$

Ponderile conexiunilor sunt modificate (în direcţia gradientului negativ) prin propagarea inversă a erorii, în scopul determinării vectorului de ponderi optim, care minimizează funcţia cost [W11, W12]:

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \eta\,\left[-\nabla C(\mathbf{w})\right] \qquad (5.34)$$

S-a notat cu η rata de învăţare. In continuare se va utiliza algoritmul de învăţare BKP [R3] cu moment (în dreapta relaţiei (5.34) se adaugă termenul $\alpha\left[\mathbf{w}(k) - \mathbf{w}(k-1)\right]$, unde α este momentul). Scopul studiului este reprezentat de etapa de proiectare "distructivă" ce urmează învăţării. Relaţiile de modificare a ponderilor în algoritmul BKP sunt prezentate în cap. 6, rescrise pentru alte funcţii cost decât norma L_2 a erorii.

5.6.2 Evaluarea performanţelor

Pentru evaluarea performanţelor structurii neurale proiectate prin aplicarea metodelor propuse, se consideră indicatorii din tab. 5.4.

❒ ***Eroarea de învăţare, eroarea de test***

Eroare pătratică medie normată la numărul patternurilor şi la numărul nodurilor de ieşire (EPMN) este evaluată atât în etapa de învăţare, cât şi în cea

de test, pe baza relaţiei (5.33).

❒ ***Raportul semnal – zgomot la vârf***

Raportul semnal – zgomot la vârf (PSNR — "Peak to peak signal to noise ratio") [P17], este definit pentru semnalul unidimensional de relaţia (5.35), unde s-au notat cu d_{max} şi d_{min} valorile maximă (5.36), respectiv minimă (5.37), ale semnalului dorit. Diferenţa acestora este valoarea la vârf.

$$\mathrm{PSNR\,[dB]} = 10\log_{10}\left[\frac{P\left(d_{max}-d_{min}\right)^2}{\sum_{\xi=1}^{P}\sum_{j=1}^{N}\left[d_j(\xi)-y_j(\xi)\right]^2}\right] = 10\log_{10}\left[\frac{\left(d_{max}-d_{min}\right)^2}{EPMN}\right] \tag{5.35}$$

$$d_{max} = \max_{1\le\xi\le P,\,1\le i\le M} d_i(\xi) \tag{5.36}$$

$$d_{min} = \min_{1\le\xi\le P,\,1\le i\le M} d_i(\xi) \tag{5.37}$$

❒ ***Valorile ponderilor obţinute***

Examinarea valorilor ponderilor obţinute se poate face numeric sau prin reprezentare grafică. O interpretare a rezultatelor procesului de învăţare este realizată în anexa 6.

Tab. 5.4 Indicatori de evaluare a performanţelor structurii neurale rezultate

Criteriul de evaluare
Eroarea pătratică medie normată la numărul de patternuri (EPMN)
Raport semnal – zgomot la vârf (PSNR [dB])
Numărul total al parametrilor structurii neurale obţinute
Valorile ponderilor obţinute
Numărul operaţiilor efectuate în etapa de test
Timpul necesar pentru etapa de test
Spaţiul necesar implementării

Scanned with OKEN Scanner

❒ *Numărul total al parametrilor structurii neurale obţinute*

Numărul total al parametrilor structurii este suma numărului ponderilor şi numărului pragurilor din reţeaua neurală. Pentru structura total interconectată, acesta se calculează cu relaţia (4.221). Pentru structurile parţial interconectate rezultate prin aplicarea algoritmilor de proiectare propuşi, devine:

$$\Pi = [(M + 1) H + (H + 1) N] (1 - \varepsilon) \qquad (5.38)$$

unde ε este procentul conexiunilor eliminate. Dacă pentru metoda OBD, în fiecare etapă de eliminare se anulează câte o singură pondere, astfel încât numărul total al acestor etape indică numărul conexiunilor tăiate din structură, acest lucru nu mai este valabil în cazul MOBD, SVD–MOBD şi TOBD.

❒ *Numărul operaţiilor de "înmulţire + adunare" efectuate în etapa de test*

Numărul operaţiilor de "înmulţire + adunare" pentru o epocă, se poate calcula în cazul structurii total interconectate cu relaţia (4.223). Pentru structurile parţial interconectate rezultate prin aplicarea algoritmilor de proiectare propuşi, aceasta se micşorează de $1 - \varepsilon$ ori, dacă ε reprezintă procentul conexiunilor eliminate.

❒ *Timpul necesar pentru etapa de test*

Evaluarea timpului necesar unei iteraţii complete BKP (o trecere "înainte" a datelor prin reţea şi o trecere "înapoi" pentru modificarea parametrilor) a fost realizată de Ghorbani şi Bhavsar [G24]. Utilizând rezultatele acestora pentru reţeaua total interconectată, se poate calcula timpul de test pentru reţeaua parţial conectată [D205].

In pasul înainte, pentru un singur nod ascuns h, $1 \le h \le H$, dintr-o reţea "feedforward" total interconectată, timpul de calcul al valorii de ieşire este $t_{c\ \text{nod ascuns}}$:

$$t_{c\ \text{nod ascuns}} = (M\ t_{*} + M\ t_{+}) + t_{f} \qquad (5.39)$$

unde s-au notat cu t_{*}, t_{+} şi t_{f} timpii necesari unei singure operaţii de înmulţire, de adunare, respectiv de calcul al valorii funcţiei neliniare de activare.

Scanned with OKEN Scanner

Dacă se presupun $t_* = t_+ = t_0$ şi numărul de ponderi mult mai mare decât numărul nodurilor, rezultă că t_f se poate neglija şi:

$$t_{c\ nod\ ascuns} \cong 2\ M\ t_0 \tag{5.40}$$

Se notează cu t_t timpul de transfer al unei date reprezentate pe b biţi, între două noduri. Rezultă că, pentru o valoare reprezentată cu precizie ridicată, timpul necesar transferului (comunicării) t_k, este un multiplu al valorii b:

$$t_k = M\ \frac{nr.biti}{b}\ t_t = \upsilon\ M\ t_t \tag{5.41}$$

Timpul de calcul pentru un nod de ieşire este dat de relaţia (5.42), iar timpul total necesar unui pas de trecere "înainte" prin reţea, de relaţia (5.43).

$$t_{c\ nod\ ieşire} = 2\ H\ t_0 \tag{5.42}$$

$$t_{înainte} = t_k + t_{c\ nod\ ascuns} + t_k + t_{c\ nod\ ieşire} = 2\ \upsilon\ M\ t_t + 2\ M\ t_0 + 2\ H\ t_0 =$$

$$= 2\ [\upsilon\ M\ t_t + (M + H)\ t_0] \tag{5.43}$$

In etapa de test se realizează o trecere "înainte" prin reţea a tuturor patternurilor, deci timpul de test este $t_{test} = t_{înainte}\ P$.

Dacă structura neurală nu este total interconectată, în expresia timpului de test, notat $t_{test\ parţial}$, în locul valorilor M, H ale dimensiunilor straturilor de intrare şi respectiv, ascuns, apare numărul efectiv de conexiuni M_{ef}, H_{ef} (relaţia (5.44)). Dacă există cel puţin un nod ascuns care este conectat la toate intrările, atunci $M_{ef} = M$, iar dacă există cel puţin un nod de ieşire conectat la toate ieşirile stratului ascuns, atunci $H_{ef} = H$. In caz contrar, valorile efective reprezintă numărul maxim de conexiuni ce intră în nodurile din stratul ascuns, respectiv stratul de ieşire, maxim evaluat pentru toate nodurile din stratul respectiv.

$$t_{test\ parţial} = 2\ [\upsilon\ M_{ef}\ t_t + (M_{ef} + H_{ef})\ t_0]\ P \tag{5.44}$$

❒ *Spaţiul necesar implementării*

Prin spaţiu necesar implementării se înţelege aria zonei necesare stocării valorilor *net*, ponderilor şi erorii elementare δ obţinute în fiecare nod al reţelei, în cazul aplicării algoritmului BKP [G24].

Scanned with OKEN Scanner

Pentru un singur nod ascuns din reţeaua total interconectată, spaţiul este:

$$sp_{nod} = M\, sp_{net} + M\, sp_w + sp_\delta \tag{5.45}$$

Dacă se presupun $sp_{net} = sp_w = sp_\delta = sp$, rezultă relaţia (5.46). Pentru implementări VLSI, cu multe noduri în chip, trebuie adăugat spaţiul necesar comunicării între acestea. Dacă se notează cu $\overline{sp}$ aria necesară unei astfel de legături, se obţine relaţia (5.47).

$$sp_{nod} = (2M + 1)\, sp \cong 2M\, sp \tag{5.46}$$

$$sp_{nod} = 2M\, sp + M\, \overline{sp} = \left(2M + M\frac{\overline{sp}}{sp}\right) sp \tag{5.47}$$

Intrucât $1 \le \upsilon \le$ *Nr. biţi*, unde *Nr. biţi* poate fi 8, 12, 16, 32, etc. (numărul ales pentru reprezentare), rezultă relaţia (5.48). Se observă că, limita inferioară şi superioară a spaţiului necesar depind de raportul dintre aria necesară interconectării şi aria memoriei în implementarea VLSI.

$$2 + \frac{\overline{sp}}{sp} \le sp_{nod} \le \left(2 + \frac{\overline{sp}}{sp}\right)(\text{Nr. biţi}) \tag{5.48}$$

Se poate arăta [G24] că, utilizând metode de comunicare incrementale între noduri (se transmit în locul valorilor parametrilor numai incrementul / decrementul faţă de valoarea anterioară), scad atât timpul de test, cât şi spaţiul necesar stocării.

Notând cu $sp_{nod,\ parţial}$ şi $\overline{sp}_{partial}$ valorile sp_{nod} şi $\overline{sp}$ pentru structura neurală parţial interconectată, relaţia (5.47) devine (5.49). Spaţiul ocupat de toate nodurile ascunse se obţine prin multiplicarea relaţiei (5.49) cu *H*.

$$sp_{nod,\ parţial} = 2M_{ef}\, sp_{parţial} + M_{ef}\, \overline{sp_{parţial}} = \left(2M_{ef} + M_{ef}\frac{\overline{sp_{parţial}}}{sp_{parţial}}\right) sp_{parţial} \tag{5.49}$$

5.6.3 Etapa de învăţare

Perceptronul multistrat a învăţat cu algoritmul BKP, un număr de 2880 de patternuri, fiecare compus din 7 eşantioane ale semnalului "chirp". Învăţarea s-a

desfăşurat pe parcursul a 2000 de epoci, cu o rată de învăţare $\eta = 0.01$ şi moment $\alpha = 0.0001$. Valoarea erorii EPMN de învăţare este 0.004456.

5.6.4 Performanţele reţelei neurale "feedforward" total interconectată, cu algorimul BKP standard

Valoarea erorii EPMN evaluată pe setul datelor de test, este 1.04526. Valorile ieşirii obţinute în faza de test şi ieşirii dorite, împreună cu eroarea pătratică medie pentru fiecare eşantion, sunt reprezentate grafic în fig. 5.3, iar detalii ale semnalelor, pentru eşantioane având indexul între 1500 şi 1575, în fig. 5.4. Numărul total al parametrilor RNA este 28 (24 ponderi + 4 praguri).

5.6.5 Performanţele reţelei neurale "feedforward" simetrică, parţial interconectată, rezultată în urma aplicării metodei MOBD

Metoda MOBD [D205, D207] este aplicată pentru reducerea structurii neurale care a învăţat la punctul 5.6.3. Ponderile având cele mai scăzute salienţe au fost eliminate, cu constrângerile de simetrie impuse în algoritmul MOBD. Valoarea finală a erorii EPMN de test, PSNR [dB] şi numărul total al parametrilor reţelei sunt prezentate în tab. 5.5, în comparaţie cu valorile anterioare aplicării metodei MOBD. Eroarea EPM pentru fiecare pattern este reprezentată în fig. 5.5.

Valorile de ieşire obţinute după 6 paşi de eliminare (5 perechi de ponderi ale conexiunilor dintre stratul de intrare – stratul ascuns şi o singură pondere dintre stratul ascuns – stratul de ieşire), precum şi valorile de ieşire dorite, sunt prezentate în fig. 5.5 şi detaliate în fig. 5.6, pentru eşantioane având index între 1500 şi 1575.

Fig. 5.7 conţine curba de reînvăţare, iar vârfurile indică paşii de eliminare a ponderilor. Valorile finale ale ponderilor conexiunilor reţelei sunt prezentate în fig. 5.8. Structura neurală obţinută după aplicarea metodei MOBD este prezentată în fig. 5.9, unde liniile punctate indică conexiunile / nodurile eliminate.

Tab. 5.5 Erori de învăţare, erori de test şi PSNR înainte şi după aplicarea algoritmului MOBD

Rezultate	Inainte de MOBD	După MOBD (9 paşi)
Eroare de învăţare	0.00445	0.002770
Eroare de test	1.04526	1.065249
PSNR [dB]	67.9976	66.8965
Nr. parametri (ponderi + praguri)	28	17

Scanned with OKEN Scanner

5.6.6 Performanţele reţelei neurale "feedforward" simetrică, parţial interconectată, rezultată în urma aplicării metodei SVD–MOBD

Reţeaua a învăţat conform precizărilor de la punctul 5.6.3. Pentru verificarea observaţiilor care au stat la baza propunerii algoritmului SVD–MOBD, este utilă examinarea corelaţiei dintre semnalele de ieşire ale nodurilor ascunse (fig. 5.10). Se observă că aceasta nu este nulă, deci există redundanţă în stratul ascuns al reţelei neurale. Corelaţia este maximă pentru nodurile ascunse 1 şi 2, precum şi pentru 1 şi 3.

Pe de altă parte, reprezentarea grafică (fig. 5.11) a covariaţiei dintre semnalul obţinut la ieşirea fiecărui nod ascuns şi semnalul de eroare la ieşirea reţelei (conform sugestiei algoritmului CASCOR) indică faptul că, modulul său este maxim pentru primul nod ascuns, deci este posibilă renunţarea la celelalte două noduri ascunse, dacă performanţele structurii nu se degradează considerabil.

Prin aplicarea descompunerii SVD asupra matricii $\boldsymbol{Z}$ a valorilor de ieşire din stratul ascuns se obţin vectorii proprii reciproc ortogonali (fig. 5.12). Din examinarea matricii diagonale obţinute prin descompunere, care conţine valorile proprii ale matricii $\boldsymbol{Z}\,\boldsymbol{Z}^{T}$, se observă că nu se poate lua o decizie privind eliminarea sau nu a unui nod ascuns [D205].

Verificarea experimentală a reducerii corelaţiei dintre nodurile ascunse, prin aplicarea transformărilor neliniare sigmoidale [O10] este realizată în fig. 5.13. Intrucât se urmăreşte obţinerea unor vectori de ponderi ortogonali, pentru nodurile ascunse (în acest caz, corelaţia este nulă), este justificată utilizarea transformării menţionată anterior, efectuată însă asupra matricii de ponderi.

După descompunerea SVD a matricii de ponderi dintre stratul ascuns şi cel de ieşire, fără modificarea matricii ponderilor dintre stratul de intrare şi cel ascuns, valoarea erorii de test EPMN a fost 1.00501, iar numărul parametrilor reţelei s-a micşorat la 10 (8 ponderi şi 2 praguri). Au fost eliminate două noduri ascunse.

Semnalul de ieşire obţinut, semnalul de ieşire dorit şi eroarea pătratică medie la fiecare epocă, sunt prezentate în fig. 5.14, iar structura neurală obţinută în fig. 5.15, unde liniile punctate indică conexiunile / nodurile eliminate.

Efectuarea reducerii matricii ponderilor conexiunilor dintre stratul de intrare şi cel ascuns prin metoda MOBD, conduce la o eroare de test EPMN şi PSNR prezentate în tab. 5.6.

Semnalul de ieşire obţinut, semnalul de ieşire dorit şi eroarea pătratică medie după această etapă, sunt prezentate grafic în fig. 5.16, iar valorile finale ale ponderilor conexiunilor reţelei şi salienţelor, în fig. 5.17. Se observă că, ponderile conexiunilor w_{41} şi w_{71} au valori egale în modul, dar cu semn schimbat, deci contribuţia lor în valoarea semnalului de ieşire din nodul ascuns

Scanned with OKEN Scanner

se anulează. De asemenea, şi salienţele lor sunt aproximativ egale. In concluzie, w_{41} şi w_{71} se pot elimina, dar se pierde simetria structurii. Pe de altă parte, ponderile w_{51} şi w_{61} au valori egale în modul, cu semn schimbat, dar salienţele lor sunt mult diferite.

Structura neurală obţinută după aplicarea metodei SVD–MOBD este prezentată în fig. 5.18, iar numărul parametrilor săi este 8 (6 ponderi şi 2 praguri).

Trebuie precizat faptul că, efectuarea descompunerii SVD asupra ambelor matrici de ponderi (ale conexiunilor dintre stratul de intrare – stratul ascuns, respectiv dintre stratul ascuns – stratul de ieşire) nu este recomandabilă, deoarece performanţele reţelei neurale se degradează considerabil (fig. 5.19).

Tab. 5.6 Erori de învăţare, erori de test şi PSNR înainte şi după aplicarea algoritmului SVD–MOBD

Rezultate	Inainte de SVD–MOBD	După SVD–MOBD
Eroare de învăţare	0.00445	0.00332
Eroare de test	1.04526	1.00588
PSNR [dB]	67.9976	67.1976
Nr. parametri (ponderi + praguri)	28	8

5.6.7 Performanţele reţelei neurale "feedforward" simetrică, parţial interconectată, cu matrice de ponderi tridiagonală, rezultată în urma aplicării metodei TOBD

Reţeaua a învăţat conform precizărilor de la punctul 5.6.3. Erorile EPMN de învăţare şi de test înainte şi după aplicarea metodei TOBD, precum şi numărul de parametri ai structurii neurale, sunt prezentate în tab. 5.7.

Tab. 5.7 Erori de învăţare, erori de test şi PSNR înainte şi după aplicarea algoritmului TOBD

Rezultate	Inainte de TOBD	După TOBD (5 paşi)
Eroare de învăţare	0.00445	0.00308
Eroare de test	1.04526	1.07343
PSNR [dB]	67.9997	65.7976
Nr. parametri (ponderi + praguri)	28	9

Scanned with OKEN Scanner

Semnalul de ieşire obţinut după 5 paşi ai TOBD, semnalul de ieşire dorit şi eroarea pătratică medie la fiecare epocă, sunt prezentate în fig. 5.20. In fig. 5.21 este reprezentată structura neurală obţinută, unde liniile punctate indică conexiunile / nodurile eliminate. Din totalul celor 15 parametri ai reţelei, trebuie stocaţi numai 9 (6 ponderi + 3 praguri), datorită simetriei valorilor elementelor matricii de ponderi.

5.6.8 Performanţele reţelei neurale "feedforward" parţial interconectată, rezultată în urma aplicării metodei OBD

In vederea comparării rezultatelor obţinute prin aplicarea metodelor propuse, s-a aplicat algoritmul OBD în varianta standard (cap. 4). După calculul salienţei fiecărei conexiuni, au fost eliminate conexiunile având ponderi cu salienţă scăzută. Eroarea EPMN de învăţare şi de test, PSNR şi numărul parametrilor structurii rezultate (18 ponderi + 4 praguri) sunt prezentate în tab. 5.8.

Valorile de ieşire obţinute după eliminarea a 6 ponderi (5 ponderi între stratul de intrare – stratul ascuns şi o pondere între stratul ascuns – stratul de ieşire), valorile de ieşire dorite şi eroarea pătratică medie pentru fiecare pattern sunt prezentate în fig. 5.22. Semnalele de ieşire pentru eşantioane având index între 1500 şi 1575, sunt desenate în fig. 5.23.

In curba de reînvăţare din fig. 5.24, vârfurile indică eliminarea ponderilor. Din examinarea numerică şi grafică rezultă că, ponderile cu valorile cele mai scăzute, nu corespund valorilor minime ale salienţelor [D205].

Structura neurală nesimetrică rezultată după aplicarea algoritmului OBD standard este reprezentată în fig. 5.25.

Tab. 5.8 Erori de învăţare, erori de test şi PSNR înainte şi după aplicarea algoritmului OBD

Rezultate	Inainte de OBD	După OBD (12 paşi)
Eroare de învăţare	0.00445	0.00339
Eroare de test	1.04526	0.98223
PSNR [dB]	67.9976	68.2971
Nr. parametri (ponderi + praguri)	28	22

Scanned with OKEN Scanner

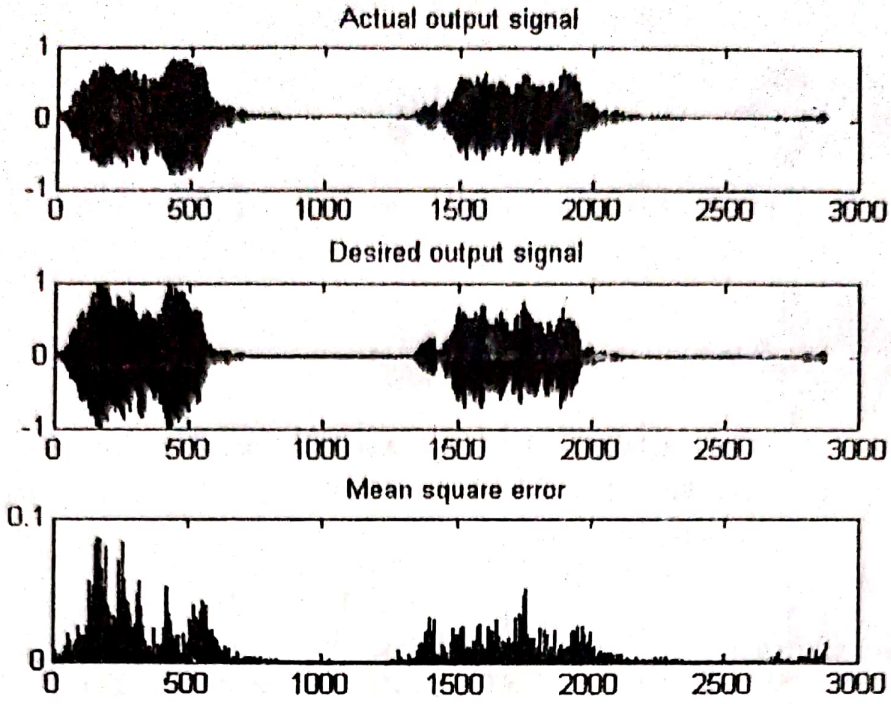


Fig. 5.3 Semnalul de ieşire obţinut, semnalul de ieşire dorit şi eroarea pătratică medie, în etapa de test

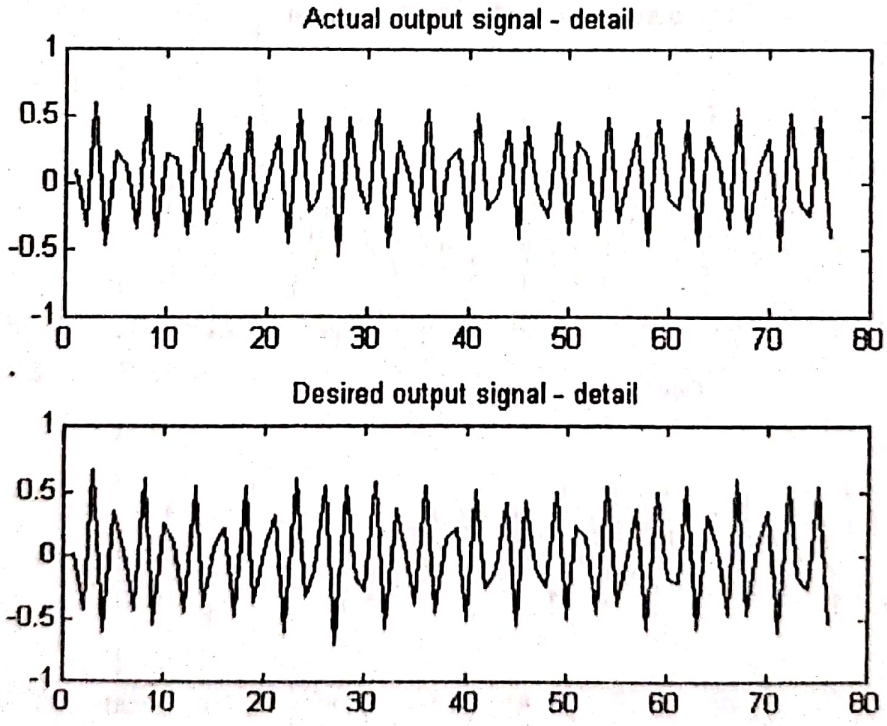


Fig. 5.4 Semnalul de ieşire obţinut şi semnalul de ieşire dorit (eşantioane cu index în intervalul 1500 – 1575)

Scanned with OKEN Scanner

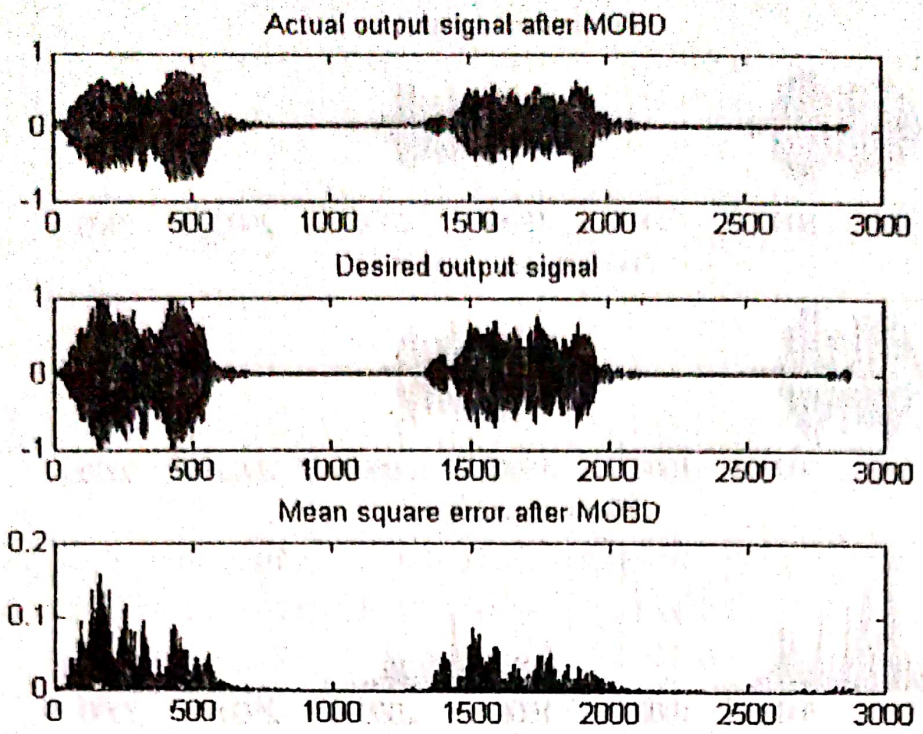


Fig. 5.5 Semnalul de ieşire obţinut, semnalul de ieşire dorit şi eroarea pătratică medie în etapa de test, după aplicarea metodei MOBD

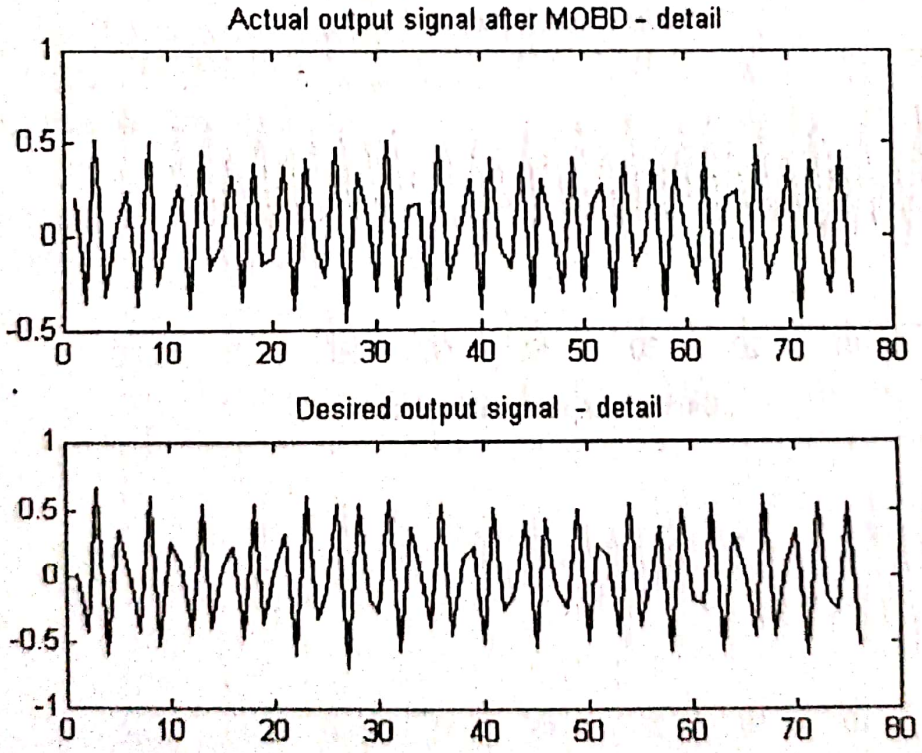


Fig. 5.6 Semnalul de ieşire obţinut după aplicarea metodei MOBD şi semnalul de ieşire dorit (eşantioane cu indexul în intervalul 1500 – 1575)

Scanned with OKEN Scanner

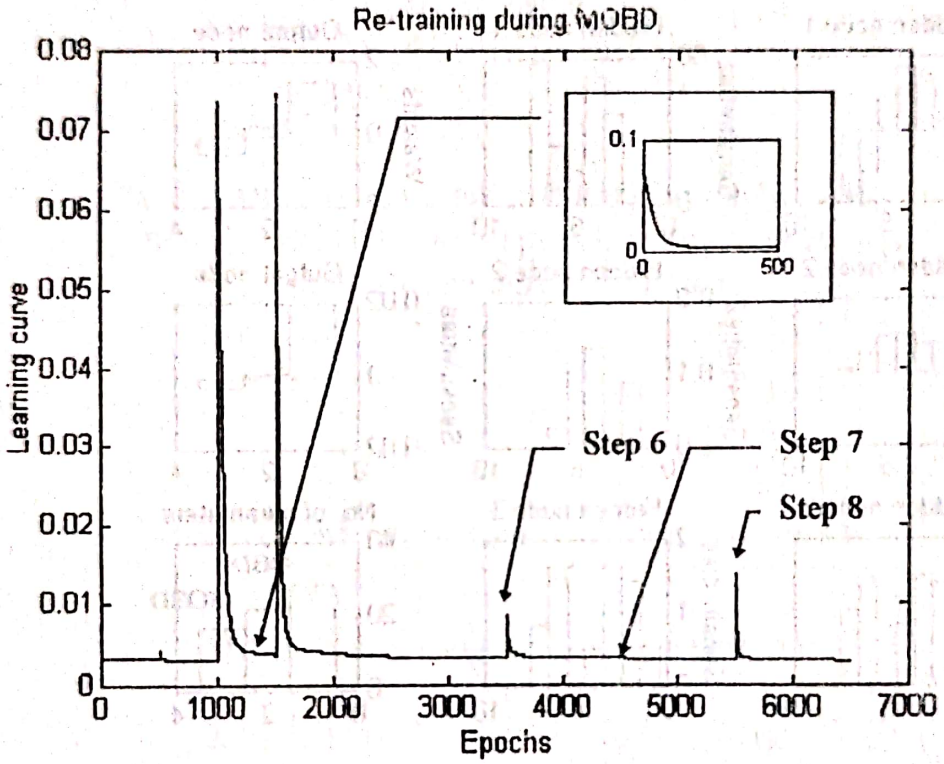


Fig. 5.7 Reînvățare după etapele de eliminare a ponderilor, în metoda MOBD. Vârfurile curbei indică eliminarea conexiunilor

Scanned with OKEN Scanner

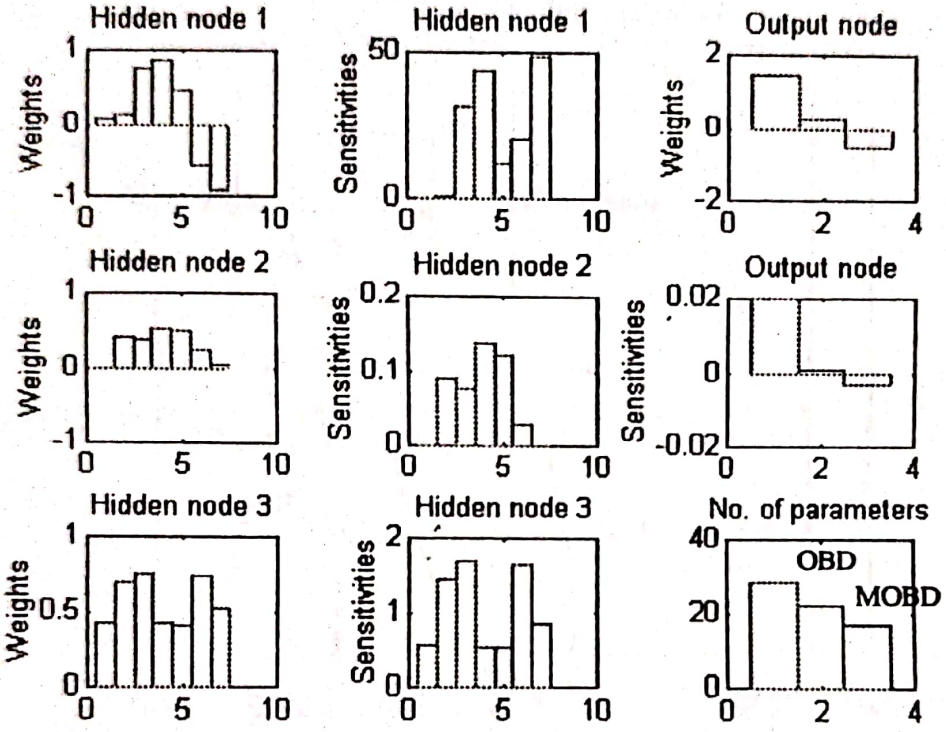


Fig. 5.8 Ponderile şi salienţele ponderilor după aplicarea metodei MOBD pentru conexiunile dintre stratul de intrare – stratul ascuns şi stratul ascuns – stratul de ieşire, precum şi numărul parametrilor reţelei

Scanned with OKEN Scanner

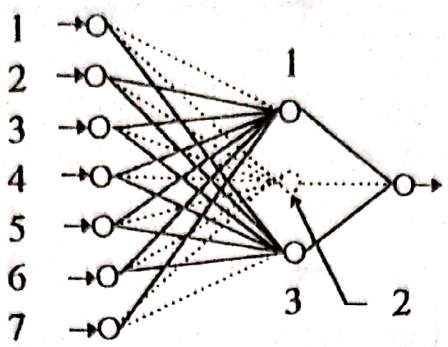


Fig. 5.9 Structura neurală rezultată prin aplicarea metodei MOBD. Liniile punctate marchează conexiunile / nodurile eliminate

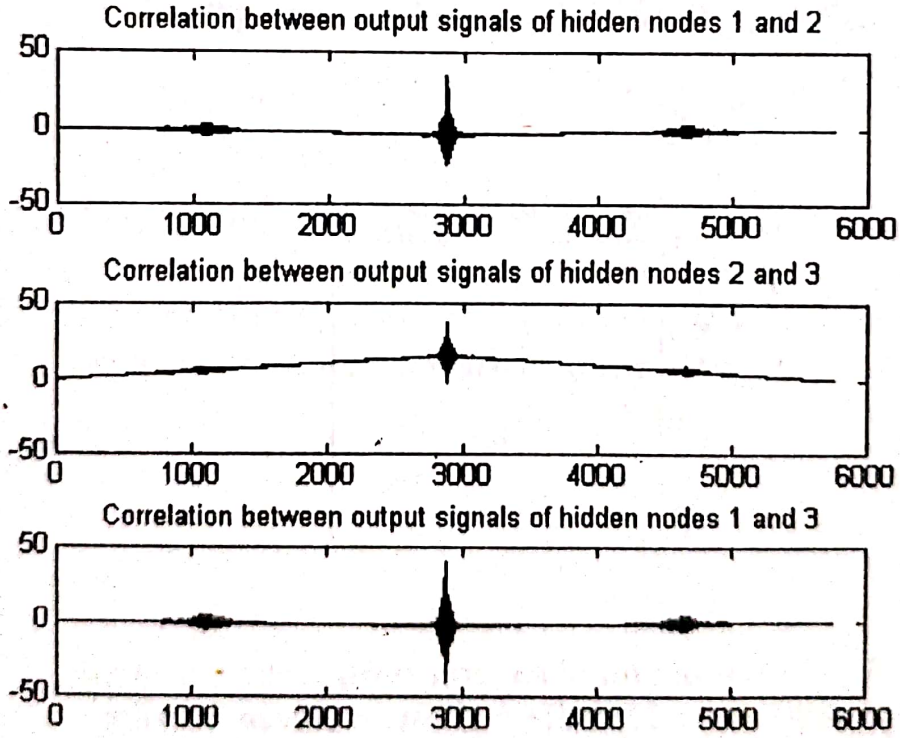


Fig. 5.10 Corelaţia între valorile de ieşire din nodurile ascunse, după etapa de învăţare

Scanned with OKEN Scanner

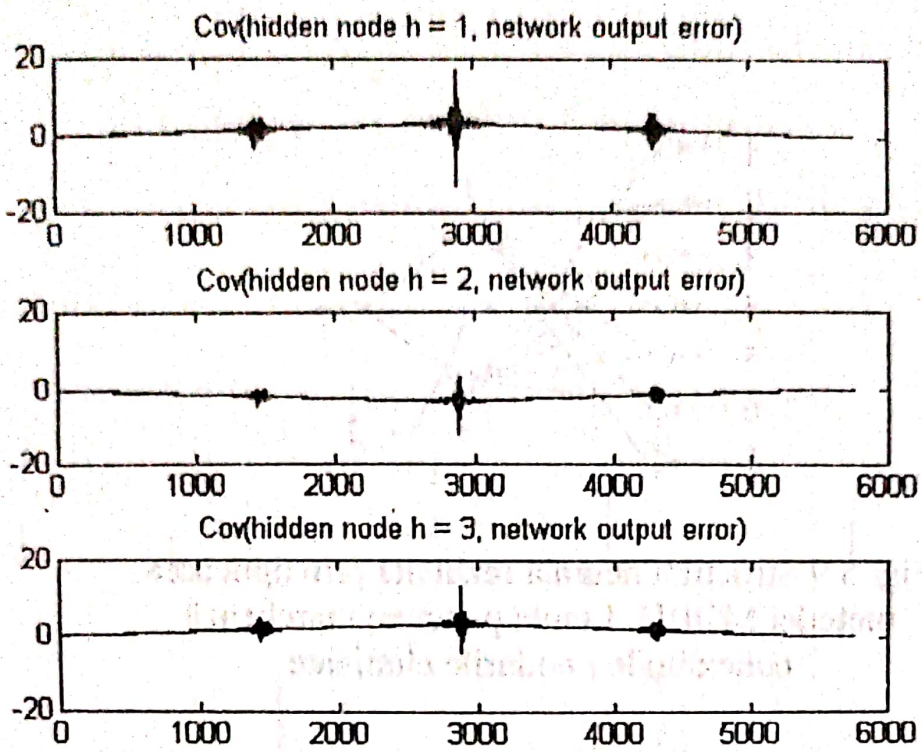


Fig. 5.11 Covariaţia dintre semnalul de ieşire din fiecare nod ascuns şi semnalul de eroare la ieşirea reţelei

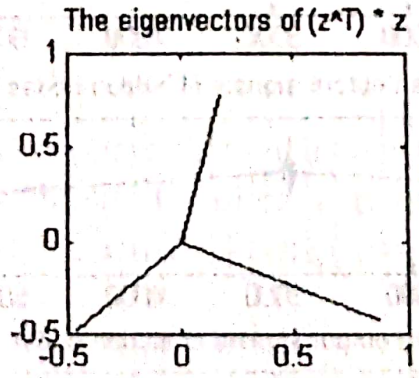


Fig. 5.12 Vectorii proprii reciproc ortogonali, ai matricii $Z^T Z$, după etapa de învăţare (Z este matricea valorilor ieşirilor din nodurile ascunse)

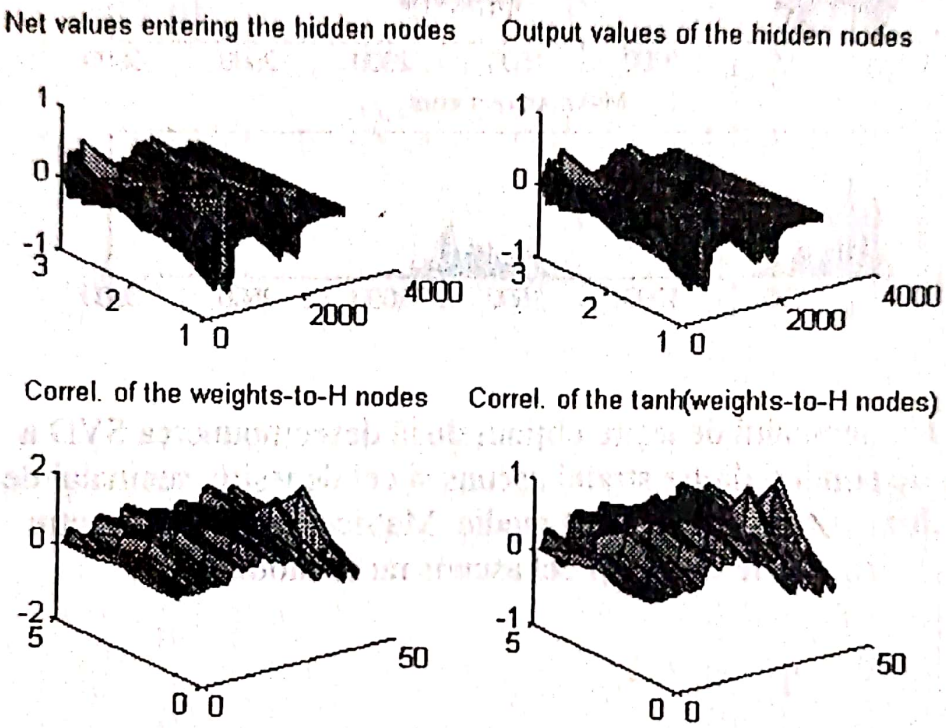


Fig. 5.13 Corelaţia valorilor *net* înainte de intrarea în stratul ascuns, corelaţia valorilor de ieşire din stratul ascuns, corelaţia valorilor ponderilor ce intră în stratul ascuns şi corelaţia valorilor ponderilor după aplicarea transformării neliniare sigmoidale

Scanned with OKEN Scanner

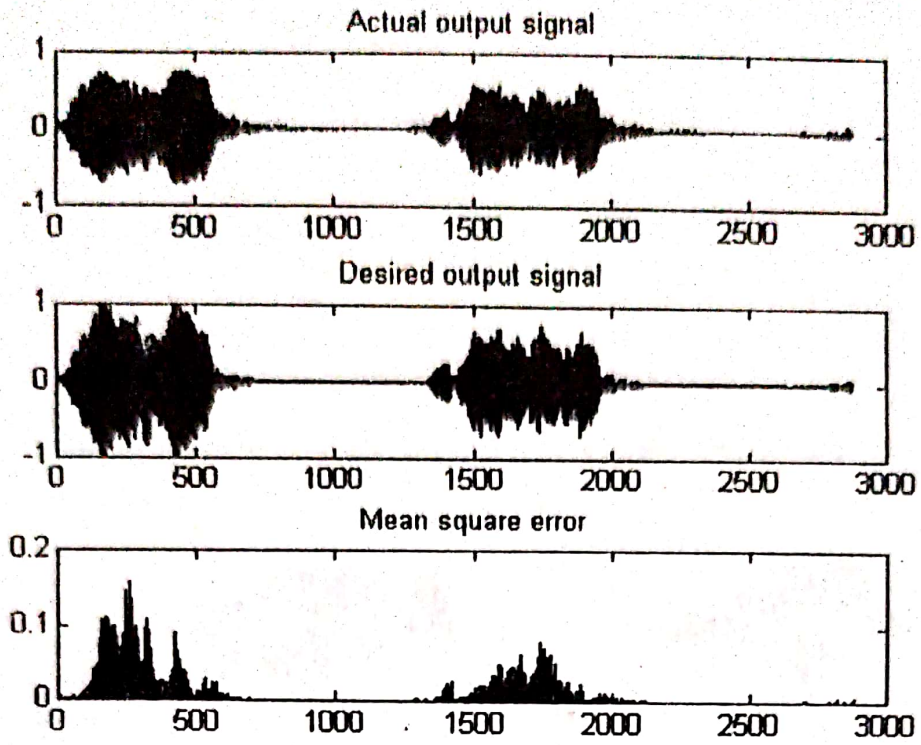


Fig. 5.14 Semnalul de ieşire obţinut după descompunerea SVD a matricii de ponderi dintre stratul ascuns şi cel de ieşire, semnalul de ieşire dorit şi eroarea pătratică medie. Matricea ponderilor dintre stratul de intrare şi cel ascuns nu se modifică

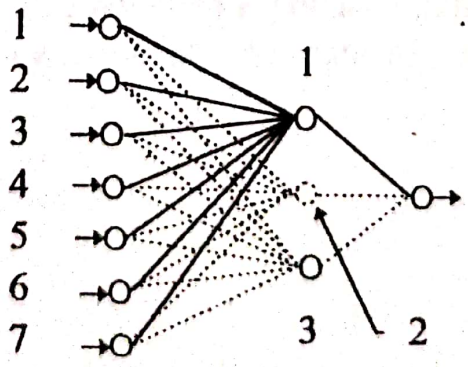


Fig. 5.15 Structura reţelei obţinute după descompunerea SVD a matricii ponderilor conexiunilor dintre stratul ascuns şi cel de ieşire

Scanned with OKEN Scanner

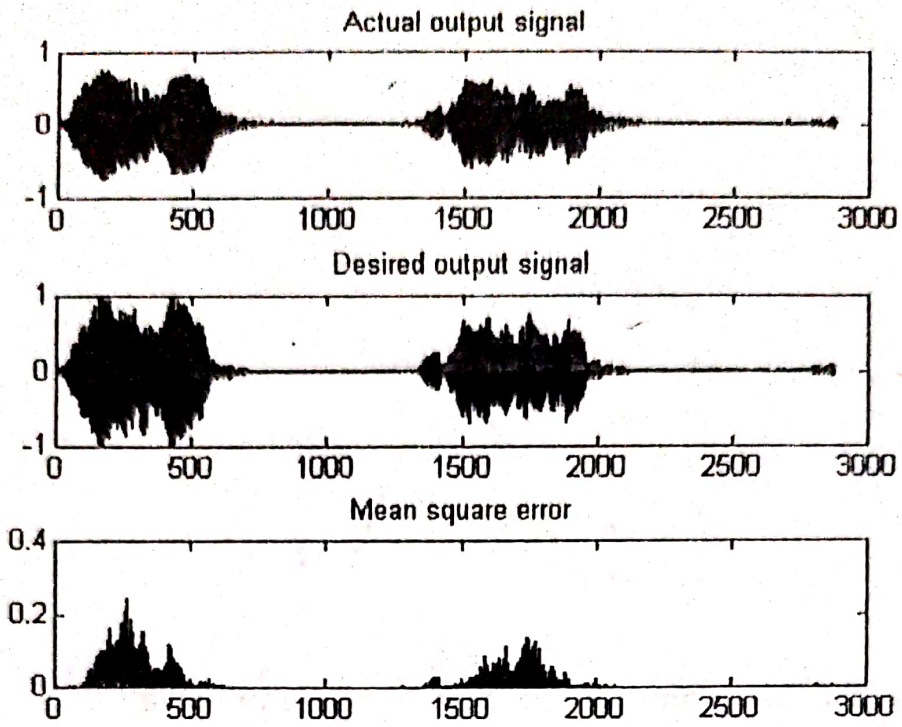


Fig. 5.16 Semnalul de ieşire obţinut de structura 5 – 1 – 1, rezultată prin aplicarea metodei SVD–MOBD, semnalul de ieşire dorit şi eroarea pătratică medie

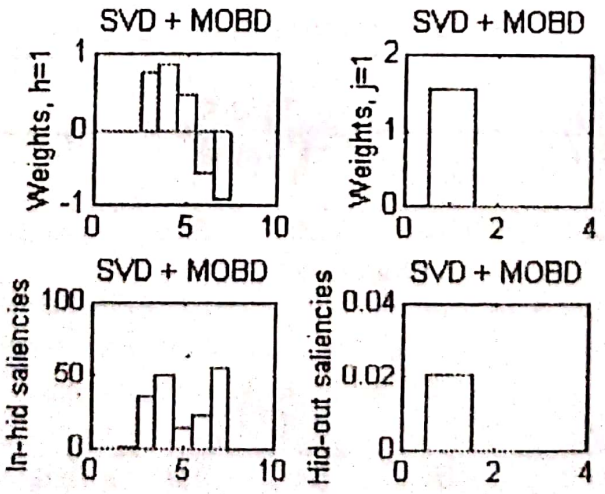


Fig. 5.17 Valorile ponderilor şi salienţelor în structura 5 – 1 – 1, obţinută prin aplicarea metodei SVD–MOBD

Scanned with OKEN Scanner

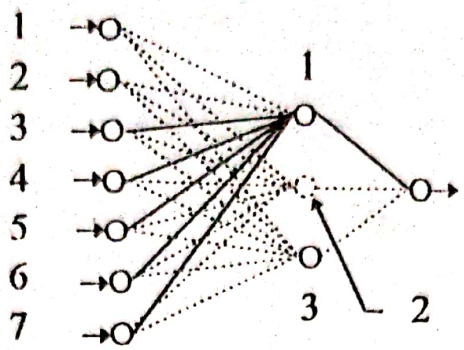


Fig. 5.18 Structura neurală 5 – 1 – 1 obţinută prin aplicarea metodei SVD–MOBD

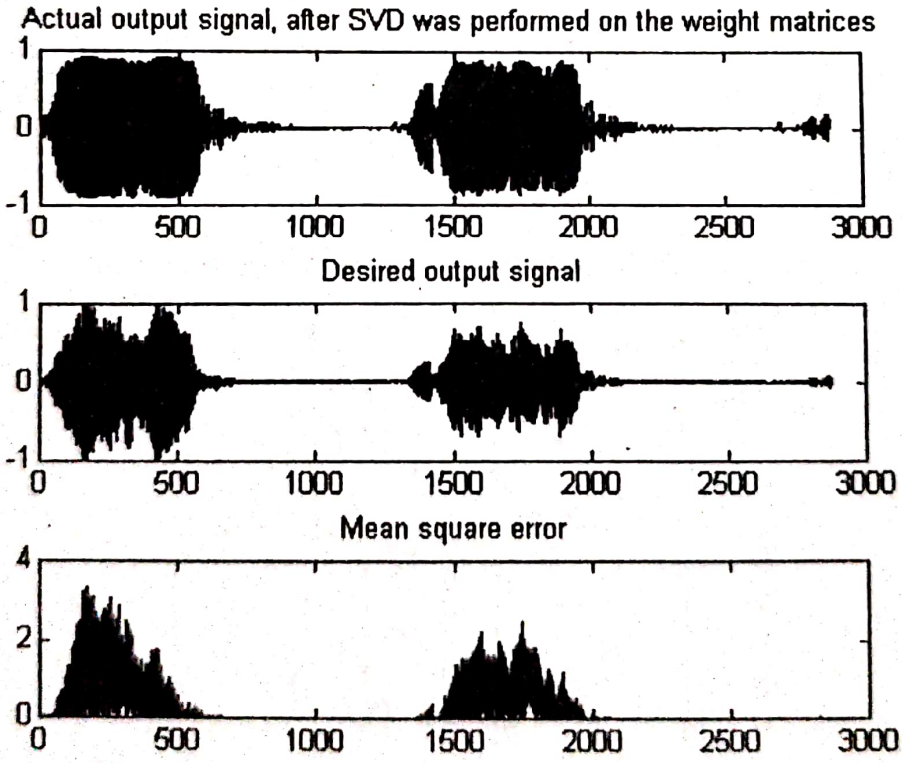


Fig. 5.19 Semnalul de ieşire obţinut şi semnalul de ieşire dorit, după realizarea descompunerii SVD a matricilor de ponderi

Scanned with OKEN Scanner

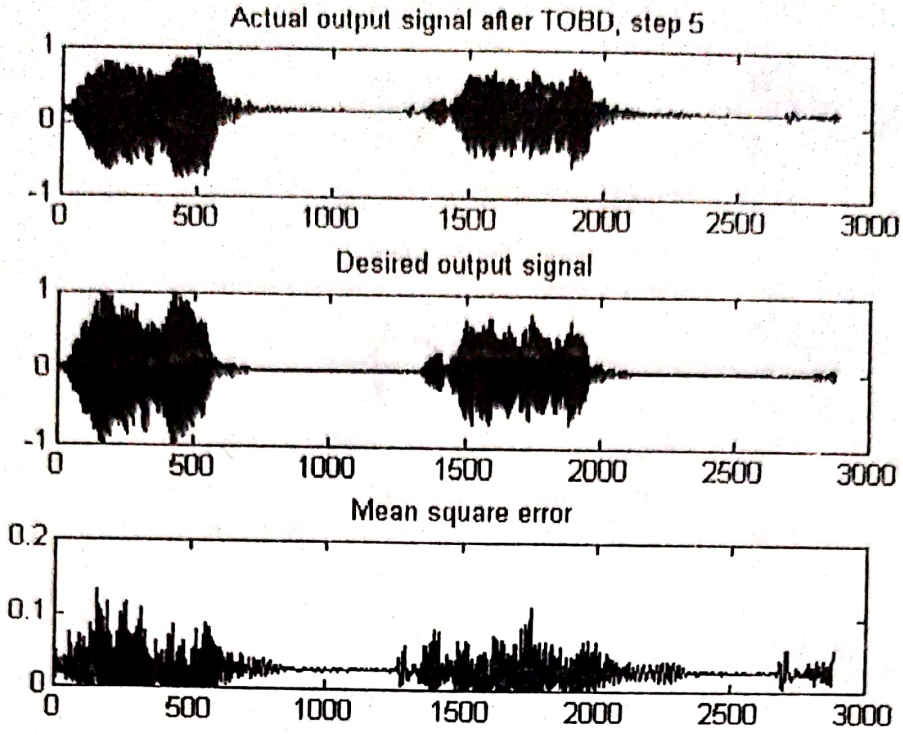


Fig. 5.20 Semnalul de ieşire obţinut, semnalul de ieşire dorit şi eroarea pătratică medie după aplicarea metodei TOBD

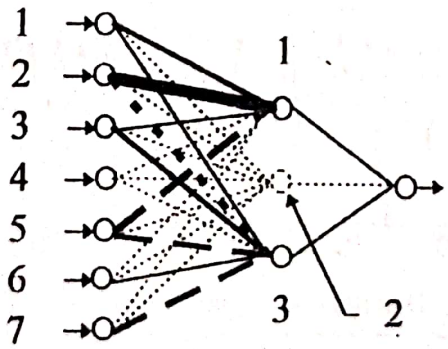


Fig. 5.21 Reţeaua neurală obţinută prin aplicarea metodei TOBD
Liniile punctate marchează conexiunile / nodurile eliminate
Conexiunile cu valori identice ale ponderilor sunt reprezentate prin linii având aceeaşi grosime. Din cele 12 ponderi ale conexiunilor, numai 7 trebuie stocate

Scanned with OKEN Scanner

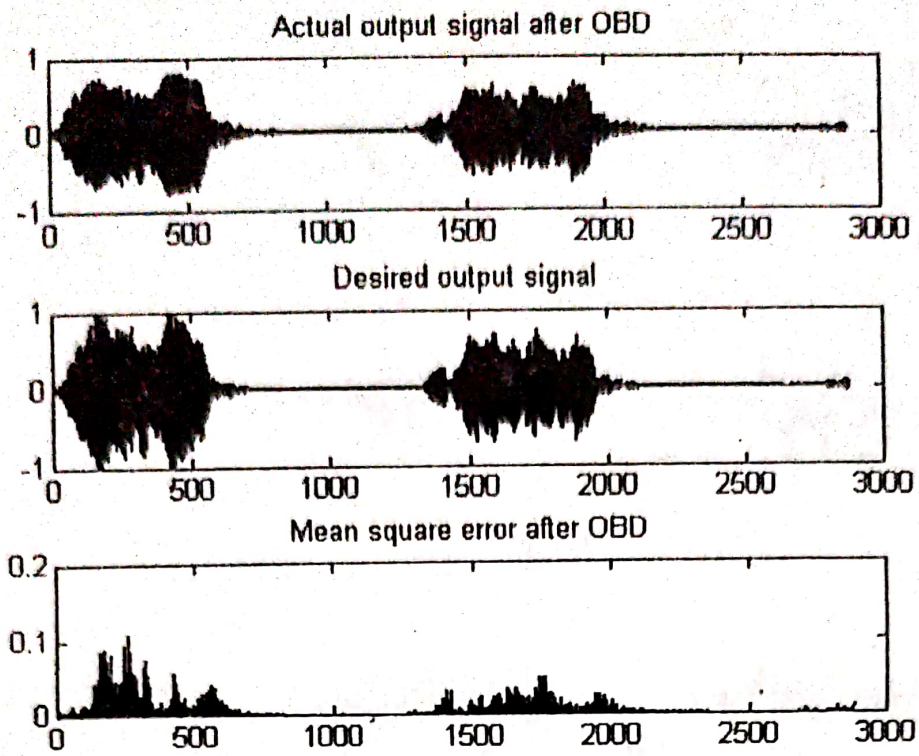


Fig. 5.22 Semnalul de ieşire obţinut, semnalul de ieşire dorit şi eroarea pătratică medie de test, după aplicarea metodei OBD standard

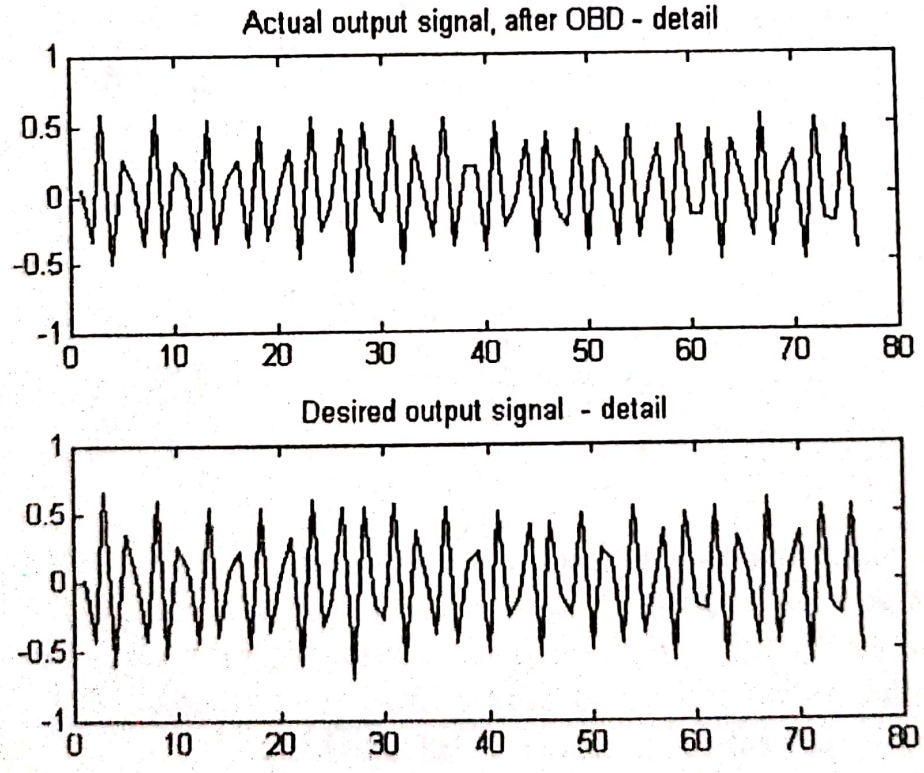


Fig. 5.23 Semnalul de ieşire obţinut după aplicarea metodei OBD standard şi semnalul de ieşire dorit (eşantioane cu indexul în intervalul 1500 – 1575)

Scanned with OKEN Scanner

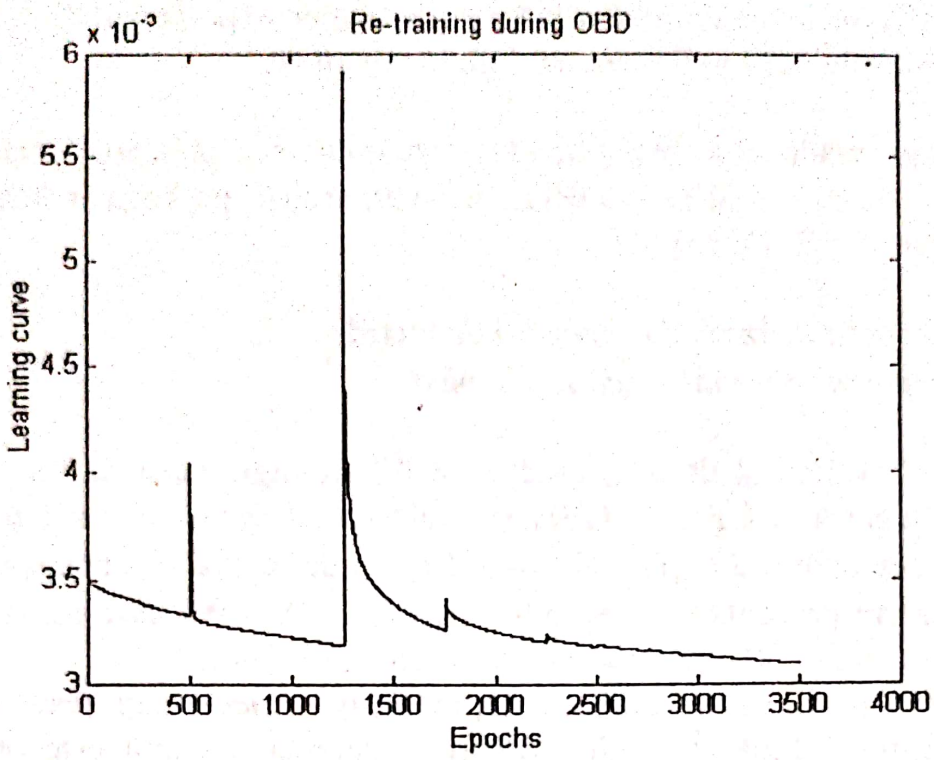


Fig. 5.24 Curba de reînvățare în metoda OBD
Vârfurile curbei indică eliminarea conexiunilor

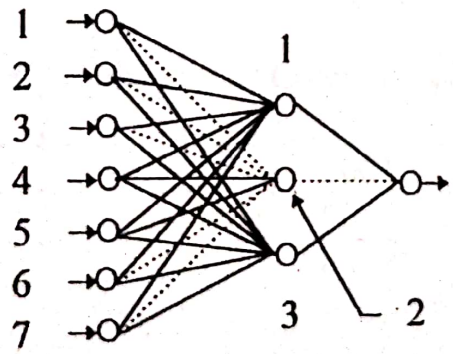


Fig. 5.25 Rețeaua neurală rezultată prin aplicarea metodei OBD. Liniile punctate marchează conexiunile / nodurile eliminate.

Scanned with OKEN Scanner

5.6.9 Interpretarea comparativă a rezultatelor experimentale obţinute prin aplicarea metodelor propuse

Comparaţia rezultatelor experimentale obţinute prin aplicarea metodelor MOBD, SVD–MOBD, TOBD şi OBD, se poate realiza pe baza indicatorilor menţionaţi în par. 5.6.2 [D205].

❒ ***Eroarea de învăţare, eroarea de test şi raportul semnal – zgomot la vârf***

Erorile de învăţare şi de test, precum şi PSNR după paşii de reducere a structurii prin algoritmii OBD, MOBD şi TOBD sunt prezentaţi în tab. 5.10. Comparaţia a fost realizată după 500 – 4000 epoci de reînvăţare, corespunzând paşilor de reducere prezentaţi în tab. 5.9. In metoda SVD–MOBD este necesar un număr de maximum 500 de epoci pentru realizarea reducerii.

Erorile EPMN de învăţare pentru toţi algoritmii menţionaţi scad faţă de valoarea înregistrată după etapa de învăţare. Diferenţa maximă este obţinută pentru metoda MOBD. Erorile de test cresc (fig. 5.26), ceea ce indică apariţia (uneori) unui proces de supraînvăţare (de exemplu: în metoda OBD, după pasul 10). Creşterea maximă a erorii de test se obţine pentru TOBD, iar cea minimă pentru SVD–MOBD.

Tab. 5.9 Numărul epocilor de reînvăţare şi pasul de reducere corespunzător, în algoritmii OBD, MOBD şi TOBD

	Epoci de reînvăţare	OBD	MOBD	TOBD
Pas de reducere a RNA FFWD	500	2	1	1
	1000	4	2	2
	2000	6	4	3
	3000	9	6	4
	4000	10	7	5

Examinarea curbelor de reînvăţare pentru MOBD şi OBD (fig. 5.7 şi 5.24), nu indică diferenţe semnificative din punct de vedere al aspectului acestor curbe. Maximele corespund eliminării câte unei ponderi din structură.

Valorile erorii de test (generalizare) evaluată cu criteriul FPE al lui Akaike, exprimat de relaţia (4.22), sunt mai mici pentru toate metodele menţionate, decât valorile obţinute pentru modelele AR(7) şi ARMAX(7) (tab. 5.11). S-au considerat valorile erorilor de învăţare şi numărul parametrilor din tabelele 5.5 – 5.8, iar $P = 2880$.

Scanned with OKEN Scanner

Tab. 5.10 Eroarea de învăţare, eroarea de test şi PSNR obţinute prin aplicarea algoritmilor MOBD, TOBD şi OBD

Nr. epocilor de reînvăţare	MOBD			TOBD			OBD		
	Eroare de învăţare	Eroare de test	PSNR [dB]	Eroare de învăţare	Eroare de test	PSNR [dB]	Eroare de învăţare	Eroare de test	PSNR [dB]
500	0.003328	1.059158	67.23	0.004371	1.073338	65.85	0.003326	1.058468	67.65
1000	0.003221	1.068139	67.23	0.003707	1.073416	62.40	0.003221	1.068183	64.95
2000	0.003949	1.066489	64.98	0.003073	1.073421	63.42	0.003257	0.829312	71.06
3000	0.003318	1.071595	64.84	0.0029061	1.073419	64.63	0.003157	0.78907	71.60
4000	0.00277	1.065249	66.89	0.0030864	1.073431	65.79	0.003394	0.98223	68.29

Obs.: Rezultatele metodei SVD–MOBD, pentru care sunt necesare numai 500 epoci de reînvăţare, au fost incluse în tab. 5.6

Scanned with OKEN Scanner

Tab. 5.11 Comparaţia modelelor rezultate prin aplicarea metodelor MOBD, SVD–MOBD, TOBD şi OBD cu modelele AR(7) şi ARMAX(7), din punct de vedere al valorii erorii de generalizare, estimată cu criteriul FPE (Akaike)

Metoda / modelul	$\hat{\Gamma}_{FPE}$
MOBD	0.0028
TOBD	0.0030
SVD–MOBD	0.0033
OBD	0.0034
Reţea total interconectată	0.0045
ARMAX(7)	0.0150
AR(7)	0.0180

Tab. 5.12 Scăderea PSNR odată cu micşorarea numărului de parametri, pentru metodele MOBD, SVD–MOBD, TOBD şi OBD

Metoda	PSNR	Numărul parametrilor structurii scade de:
MOBD	Scade cu 1.079 dB	1.65 ori
SVD–MOBD	Scade cu 0.800 dB	3.50 ori
TOBD	Scade cu 2.200 dB	3.11 ori
OBD	Creşte cu 0.290 dB	1.27 ori

Raportul semnal – zgomot la vârf exprimat în decibeli este afectat de reducerea parametrilor structurii prin metodele propuse (tab. 5.12). In toate cazurile, cu excepţia algoritmului OBD, PSNR scade, cu valori între minimum 0.8 dB (SVD–MOBD) şi maximum 2.2 dB (TOBD), pentru reduceri ale numărului parametrilor de peste 3 ori. Valoarea sa se înrăutăţeşte cel mai puţin în cazul metodei SVD–MOBD, în care sunt necesare numai 500 de epoci de reînvăţare, faţă de celelalte metode. De asemenea, pentru aproximativ aceeaşi reducere a numărului parametrilor, PSNR de modifică substanţial în metoda TOBD, faţă de SVD–MOBD.

Evaluarea PSNR după un număr mult mai mare de epoci de reînvăţare (peste 10 000) indică însă creşterea sa peste nivelul raportului semnal – zgomot al structurii total interconectate care a învăţat, acest rezultat fiind în concordanţă cu observaţiile care, indică o creştere a capacităţii de generalizare, odată cu

reducerea numărului de parametri ai structurii.

❒ *Valorile ponderilor obţinute*

Valorile ponderilor dintre stratul de intrare şi cel ascuns, rezultate din MOBD şi OBD, sunt prezentate în fig. 5.27, iar cele dintre stratul ascuns şi cel de ieşire, în fig. 5.28. Comparaţia a fost realizată pentru acelaşi număr de epoci de reînvăţare.

Compararea valorilor ponderilor, reprezentate la fiecare pas al algoritmilor TOBD şi OBD indică faptul că, nu există diferenţe semnificative din punct de vedere al domeniului valorilor ponderilor [D205]. Cu toate acestea, TOBD conduce la menţinerea conexiunilor cu valori ale simetrice ale ponderilor, în structură.

❒ *Numărul total al parametrilor structurii neurale obţinute*

Numărul total al parametrilor reţelei obţinute prin metodele MOBD, SVD–MOBD, TOBD şi OBD este prezentat în tab. 5.13, comparativ cu cel al reţelei iniţiale total interconectate. Metoda SVD–MOBD conduce la numărul minim de parametri, urmată de TOBD şi MOBD.

Structurile neurale rezultate ca urmare a aplicării algoritmilor MOBD, SVD–MOBD, TOBD şi OBD din fig. 5.8, 5.17, 5.21 şi 5.25. Reţeaua obţinută prin metoda OBD poate fi simplificată în continuare, continuând reducerea acesteia. Structurile MOBD şi TOBD sunt simetrice, dar pentru cea din urmă, trebuie stocaţi numai 7 dintre cei 12 parametri (ponderi), întrucât valorile obţinute sunt simetrice. Cel de-al doilea nod ascuns a fost eliminat în fiecare dintre cele trei cazuri.

Tab. 5.13 Numărul ponderilor, pragurilor şi numărul total al parametrilor ce trebuie stocaţi după aplicarea metodelor SVD–MOBD, TOBD, MOBD şi OBD

Algoritm	Numărul ponderilor	Numărul pragurilor	Numărul total al parametrilor ce trebuie stocaţi
SVD–MOBD	6	2	8
TOBD	12	3	9
MOBD	14	3	17
OBD	18	4	22
Reţeaua total interconectată	24	4	28

Scanned with OKEN Scanner

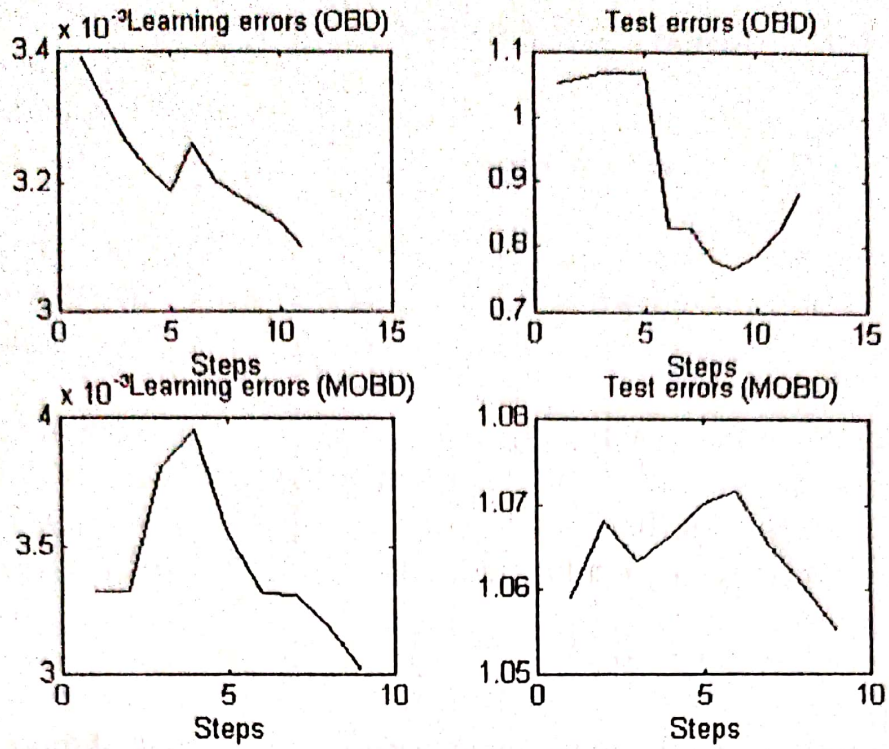


Fig. 5.26 Erorile de învăţare şi de test obţinute pentru metodele OBD şi MOBD

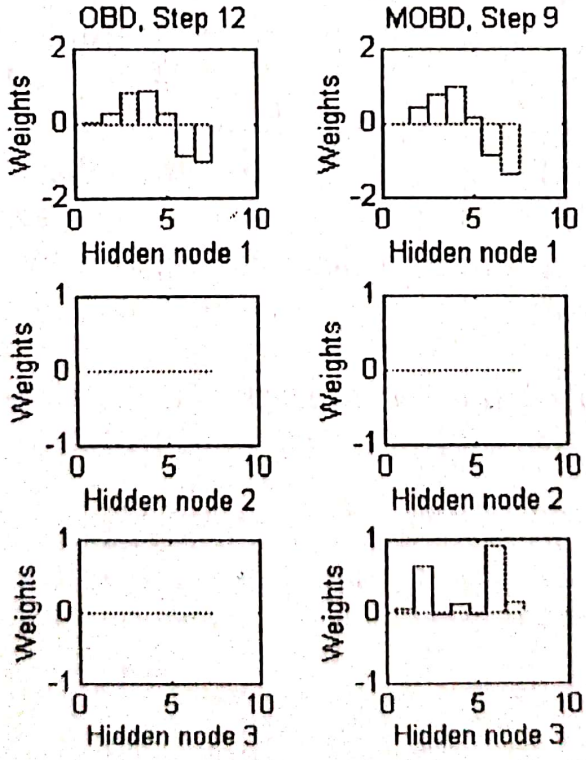


Fig. 5.27 Comparaţie între ponderile conexiunilor dintre nodurile stratului de intrare şi nodurile stratului ascuns, obţinute prin aplicarea metodelor MOBD şi OBD

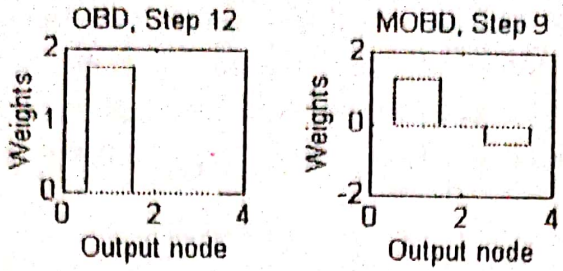


Fig. 5.28 Comparaţie între ponderile conexiunilor dintre nodurile stratului ascuns şi nodul de ieşire, obţinute prin aplicarea metodelor MOBD şi OBD

Criteriul FPE al lui Akaike, exprimat de relaţia (4.22), aplicat pentru diferite valori ale raportului între eroarea de generalizare şi eroarea de învăţare şi un număr de patternuri egal cu cel al ferestrelor adiacente de dimensiune egală cu 7, suprapuse peste semnal (2880 / 7 = 414) conduce la un estimat al numărului parametrilor totali $\hat{\Pi}_{FPE}$ de valoare foarte mare (tab. 5.14). După cum am menţionat anterior, alegerea acestuia conform criteriului FPE determină supraînvăţarea.

Tab. 5.14 Estimatul numărului parametrilor, conform criteriului FPE

$\hat{\Gamma}_{FPE} / C_{înv}$	$\hat{\Pi}_{FPE}$
1.1	19.72
1.2	37.63
2	138
10	338
100	405
320	411.42
350	411.64

Aplicând relaţia Baum – Haussler – Widrow (4.5), pentru determinarea numărului total al parametrilor structurii, pentru $P = 2880$ şi $\varepsilon = 0.01$, rezultă $\Pi < 10$. Conform tab. 5.13, numai rezultatele obţinute de algoritmii SVD–MOBD şi TOBD îndeplinesc această condiţie.

Este interesant faptul că, deşi numărul total al parametrilor reţelei descreşte pentru metodele propuse, numărul parametrilor efectivi creşte în timpul reînvăţării. La începutul procesului de reînvăţare, parametrii sunt iniţializaţi cu

valori pentru care nu se obţine o valoare minimă a funcţiei cost. Deşi toţi parametrii sunt prezenţi, totuşi numărul parametrilor efectivi este aproape nul, pentru că nu au încă nici o contribuţie utilă în rezolvarea problemei date. Pe măsură ce se desfăşoară procesul de reînvăţare, numărul parametrilor efectivi creşte [W31].

❒ *Numărul operaţiilor de "înmulţire + adunare" efectuate în etapa de test*

Numărul operaţiilor de "înmulţire + adunare" efectuare în etapa de test este prezentat în tab. 5.15. Se observă o scădere a acestuia de 10.5 ori pentru metoda SVD–MOBD, de 4.66 ori pentru TOBD, de 2.47 ori pentru MOBD şi de 1.90 ori pentru OBD, faţă de numărul operaţiilor amintite pentru reţeaua neurală total interconectată.

Tab. 5.15 Numărul operaţiilor de "înmulţire + adunare" în etapa de test, pentru structurile rezultate prin metodele MOBD, SVD–MOBD, TOBD şi OBD

Structura rezultată prin metoda	Numărul operaţiilor de "înmulţire + adunare" în etapa de test
SVD–MOBD	6582
TOBD	14810
MOBD	27975
OBD	36205
Reţea total interconectată	69120

❒ *Timpul necesar pentru etapa de test*

Duratele etapelor de testare pentru fiecare dintre metodele propuse, în comparaţie cu duratele de test calculate cu relaţia (5.44), sunt incluse în tab. 5.16. Evaluarea s-a realizat pe un calculator IBM – PC 486 / 66 MHz.

S-a notat cu t_0 durata unei operaţii simple de adunare sau înmulţire, cu t_t timpul de transfer al unei date reprezentate pe $b = 16$ biţi. S-a presupus un raport υ unitar între numărul biţilor folosiţi pentru reprezentare şi valoarea b, iar $P = 2880$.

Dacă se consideră ca referinţă durata etapei de test pentru structura rezultată prin aplicarea metodei SVD–MOBD, rezultă că pentru RNA obţinute din TOBD, MOBD, OBD şi pentru RNA iniţială sunt necesare, respectiv, cu

Scanned with OKEN Scanner

5760 t_0, cu 5760 ($t_t + t_0$), cu 5760 ($2\ t_t + 3\ t_0$) şi cu 5760 ($2t_t + 4\ t_0$) mai multe unităţi de timp.

Tab. 5.16 Duratele etapelor de testare

Structura rezultată prin metoda	Durata calculată a etapei de test	Durata măsurată a etapei de test [s]
SVD–MOBD	5760 ($5\ t_t + 6\ t_0$)	10
TOBD	5760 ($5\ t_t + 7\ t_0$)	15
MOBD	5760 ($6\ t_t + 7\ t_0$)	22
OBD	5760 ($7\ t_t + 9\ t_0$)	25
Reţeaua total interconectată	5760 ($7\ t_t + 10\ t_0$)	42

❒ ***Spaţiul necesar implementării***

Spaţiul necesar implementării fiecăreia dintre structurile neurale rezultate pentru metodele propuse, este aproximativ spaţiul necesar stratului ascuns, calculat prin multiplicarea relaţiei (5.49) cu *H*. Rezultatele obţinute sunt incluse în tab. 5.17. S-a folosit notaţia (5.50), unde $sp_{\text{parţial}}$, $\overline{sp}_{\text{parţial}}$ au semnificaţia precizată în par. 5.6.2.

$$\text{sp}_{\text{reţea, parţial (SVD–MOBD)}} = 5\left(2 + \frac{\overline{\text{sp}_{\text{parţial}}}}{\text{sp}_{\text{parţial}}}\right) \text{sp}_{\text{parţial}}\ \text{H} \qquad (5.50)$$

Spaţiul de implementare minim (considerat referinţă) este necesar structurii rezultate din metoda SVD–MOBD. Algoritmii MOBD, TOBD şi OBD conduc la reţele care ocupă un spaţiu de 1.2, 2, respectiv 2.8 ori mai mare decât cel de referinţă.

Tab. 5.17 Spaţiul necesar implementării structurilor neurale rezultate

Algoritm	Spaţiul necesar implementării (calculat)
SVD–MOBD	$\text{sp}_{\text{reţea, parţial (SVD–MOBD)}}$ (referinţă)
TOBD	2 $\text{sp}_{\text{reţea, parţial (SVD–MOBD)}}$
MOBD	1.2 $\text{sp}_{\text{reţea, parţial (SVD–MOBD)}}$
OBD	2.8 $\text{sp}_{\text{reţea, parţial (SVD–MOBD)}}$
Reţea total interconectată	4.2 $\text{sp}_{\text{reţea, parţial (SVD–MOBD)}}$

Scanned with OKEN Scanner

5.7 CONCLUZII, DIRECŢII DE STUDIU ŞI PERSPECTIVE

Scopul urmărit a fost introducerea constrângerilor de simetrie structurală în RNA. Alegerea modelului "feedforward" cu învăţare supervizată este motivată atât de faptul că, studiul acestui tip de reţea reprezintă subiectul lucrării, cât şi de faptul că, reţeaua menţionată are proprietatea de aproximare universală (cap. 3.2.2).

Regresia neliniară a fost aleasă ca problemă de studiu, pentru cazul unui semnal real. Intrucât problemele de clasificare pot fi abordate ca fiind cazuri particulare ale celor de regresie, concluziile obţinute pot fi extinse.

5.7.1 Concluzii teoretice

Metodele de proiectare "distructivă" cu constrângeri de simetrie, incluse în acest capitol, au fost următoarele:

❶ Metoda MOBD în care, la fiecare pas, este eliminată atât ponderea conexiunii cu salienţă minimă calculată conform metodei OBD, cât şi "perechea" sa simetrică din punct de vedere al poziţiei în reţea.

❷ Metoda SVD–MOBD, bazată pe descompunerea SVD a matricii ponderilor conexiunilor dintre stratul ascuns şi stratul de ieşire, pe baza căreia se elimină noduri ascunse. Este urmată de simetrizarea matricii ponderilor conexiunilor dintre stratul de intrare şi cel ascuns, pe baza unui pas MOBD.

❸ Metoda TOBD, care conduce la matrici de ponderi tridiagonale, prin efectuarea unei transformări de reflexie Householder asupra matricii ponderilor dintre stratul de intrare şi cel ascuns. In acest scop, au fost deduse relaţiile matematice de calcul, precizând şi modalităţile de stocare ale elementelor matricii rezultate.

Concluzia teoretică este aceea că, pentru toţi cei trei algoritmi, obţinerea simetriei se realizează cu preţul creşterii erorii, deşi există o limită superioară a valorii acesteia.

5.7.2 Concluzii experimentale

Studiul experimental realizat a fost constituit din următoarele etape:

❶ Formularea problemei de studiu (regresie neliniară) şi precizarea modului de construire a seturilor de date de învăţare şi de test, pe baza

Scanned with OKEN Scanner

semnalului real "chirp" disponibil.

❷ Alegerea structurii de reţea "feedforward" perceptron multistrat, cu dimensiunile 7 – 3 – 1, utilizând relaţia Baum – Haussler – Widrow (4.8).

❸ Evaluarea erorii de învăţare şi a PSNR după etapa de învăţare supervizată cu algoritmul BKP cu moment, precum şi după testarea reţelei pe setul datelor de test.

❹ Aplicarea fiecăreia dintre metode (MOBD, SVD–MOBD, TOBD) asupra reţelei care a învăţat şi evaluarea indicatorilor de performanţă; reprezentarea grafică a structurii neurale finale, a valorilor de ieşire obţinute şi a valorilor de ieşire dorite.

❺ Pentru realizarea comparaţiei, aplicarea metodei OBD asupra reţelei care a învăţat şi evaluarea indicatorilor de performanţă menţionaţi; reprezentarea grafică a structurii neurale finale, a valorilor de ieşire obţinute şi a valorilor de ieşire dorite.

❻ Compararea cantitativă a rezultatelor experimentale obţinute prin aplicarea metodelor MOBD, SVD–MOBD, TOBD şi OBD, şi de asemenea, cu cele ale modelelor AR(7) şi ARMAX(7).

S-a constatat că, eroarea de învăţare scade după aplicarea tuturor algoritmilor menţionaţi, faţă de valoarea înregistrată după etapa de învăţare. Diferenţa maximă este obţinută pentru structura rezultată prin metoda MOBD. Erorile de test cresc după aplicarea tuturor metodelor menţionate, situându-se însă sub valorile obţinute pentru modelele AR(7) şi ARMAX(7).

PSNR scade pentru toate metodele menţionate, excepţie făcând OBD. Din acest punct de vedere, SVD–MOBD şi MOBD obţin rezultate similare. Examinarea scăderii numărului de parametri în paralel cu cea a valorilor PSNR, indică performanţele cele mai bune pentru SVD–MOBD (scădere minimă a PSNR pentru scădere maximă a numărului parametrilor) şi cele mai slabe pentru TOBD (scădere mare a valorii PSNR pentru scădere maximă a numărului parametrilor).

Numărul total al parametrilor reţelelor rezultate este minim pentru cea rezultată din SVD–MOBD, urmată de TOBD şi MOBD. Deşi toate conduc la structuri parţial interconectate simetrice, comparaţia sa cu estimatul numărului parametrilor, determinat cu criteriul FPE al lui Akaike, indică faptul că, toate valorile obţinute sunt mai mici decât cele determinate teoretic.

Examinarea valorilor ponderilor rezultate arată că, deşi domeniul valorilor ponderilor este aproximativ acelaşi, pentru toate structurile rezultate, deseori în urma aplicării algoritmilor apar ponderi cu valori simetrice. De aceea, numărul parametrilor stocaţi în final este mai redus decât numărul total al conexiunilor păstrate în reţea.

Scanned with OKEN Scanner

Numărul operaţiilor de "înmulţire + adunare" este minim pentru reţeaua obţinută prin aplicarea metodei SVD–MOBD, urmată de TOBD, MOBD şi OBD.

Evaluarea timpului necesar pentru etapa de test în structurile neurale obţinute, utilizând un calculator IBM PC 486/66 MHz şi compararea rezultatelor obţinute cu cele teoretice, conduce la următoarea ierarhie: SVD–MOBD (timp minim), TOBD, MOBD, OBD, reţea total interconectată (timp maxim).

Spaţiul calculat, necesar implementării fiecăreia dintre structurile neurale obţinute, este minim pentru reţeaua obţinută cu metoda SVD–MOBD, urmată de MOBD, TOBD, OBD şi structura total interconectată.

Comparaţiile experimentale realizate confirmă concluzia teoretică. Dintre metodele propuse, compromisul "optim" între indicatorii de evaluare este realizat de SVD–MOBD, care în plus, necesită un timp de reînvăţare minim.

Deşi metoda TOBD conduce la rezultate bune din punct de vedere al dimensiunilor structurii, deci şi timpului de test şi spaţiului de implementare necesar, PSNR scade semnificativ şi este necesar un timp îndelungat de reînvăţare pentru îmbunătăţirea performanţelor. Se poate considera că, metodele TOBD şi MOBD, sunt la fel de performante, cea din urmă bazându-se pe aproximaţii mai drastice, dar având o complexitate redusă a etapei de proiectare.

5.7.3 Direcţii de studiu şi perspective

❶ Imbunătăţirea performanţelor metodelor propuse se poate realiza prin renunţarea la aproximarea diagonală a matricii hessiene, care stă la baza calculului salienţelor în algoritmul OBD. Utilizarea întregii matrici hessiene măreşte precizia, dar complexitatea calculului, deci şi consumul de resurse (timp, spaţiu de stocare) cresc semnificativ.

❷ Utilizarea altor funcţii cost decât norma L_2 a erorii, pentru caracterizarea modelului datelor (cap. 6) şi a algoritmilor de ordin doi pentru etapele de reînvăţare reprezintă soluţii de reducere a numărului de epoci necesar revenirii la valori PSNR dorite. Deşi algoritmul BKP cu moment aplicat în studiile experimentale, utilizează valori ale ponderilor din pasul precedent, viteza sa şi precizia de determinare a minimului funcţiei cost sunt inferioare celor ale algoritmilor de ordin doi amintiţi în par. 3.2.2. Pe de altă parte, utilizarea acestora creşte spaţiul de stocare necesar parametrilor.

❸ Existenţa unui număr mare de simulatoare din categoria "free software" sau "public software", pe calculatoare puternice, face posibilă parcurgerea rapidă a etapei de învăţare, pentru rezolvarea problemelor cu RNA "feedforward". Reducerea structurii neurale care a învăţat, prin aplicarea algoritmilor propuşi, reprezintă o etapă care poate urma învăţării, structurile simetrice obţinute făcând

Scanned with OKEN Scanner

posibilă extragerea regulilor necesare unui sistem expert. La interpretarea rolului nodurilor rămase în structură contribuie şi precizările incluse în anexa 6.

❹ O problemă importantă de perspectivă o constituie continuarea studiului teoretic al problemei de reprezentare, pentru stabilirea limitelor în care o structură neurală "feedforward" parţial interconectată poate aproxima o relaţie intrare – ieşire.

5.8 REZUMATUL CAPITOLULUI

După motivarea necesităţii introducerii constrângerilor de simetrie în proiectarea structurilor neurale "feedforward", în cadrul capitolului au fost prezentate trei metode "distructive", bazate pe reducerea reţelei neurale după etapa de învăţare. Pentru problema de studiu (regresie neliniară), s-a propus rezolvarea cu ajutorul structurii neurale "feedforward" perceptron multistrat. Au fost precizate: construirea seturilor de date pe baza semnalului real considerat, dimensionarea reţelei neurale, performanţele reţelei în etapa de învăţare şi de test. Au fost realizate comparaţii ale rezultatelor teoretice şi experimentale şi au fost formulate concluzii, identificându-se problemele încă nerezolvate, precum şi perspectivele studiului.

Scanned with OKEN Scanner

6 PROIECTAREA REȚELELOR NEURALE "FEEDFORWARD" CU FUNCȚII COST DIFERITE DE NORMA L_2 A ERORII

Obținerea unor performanțe bune ale RNA în etapele de învățare și de test în general, precum și în etapele de reînvățare din metodele de proiectare cu constrângeri de simetrie în particular, este condiționată de alegerea modelului datelor. Am precizat în par. 3.1.3 că, aplicația ce trebuie rezolvată cu structuri neurale "feedforward" este caracterizată de datele disponibile, care determină alegerea unui criteriu (funcție) cost, în conformitate cu care rețeaua învață și este evaluată. Funcția cost nu caracterizează direct modelul neural propriu – zis, ci reprezintă o anumită ipoteză asupra distribuției datelor de intrare, mai precis asupra modelelor de zgomot aleator existent în datele de intrare. Cunoașterea apriori a acestei distribuții, sau existența datelor empirice care să permită estimarea sa fac posibilă alegerea funcției cost [W29, C16, C43], depinzând însă și de tipul problemei ce trebuie rezolvată (estimare, clasificare, etc.).

6.1 ALEGEREA CRITERIULUI DE COST ÎN FUNCȚIE DE DISTRIBUȚIA ERORILOR ÎN SETUL DE DATE

Presupunem că funcția cost este aditivă, dată de relația (3.23). S-a notat cu P numărul maxim al patternurilor de învățare prezentate la intrarea rețelei, indicele fiecărui pattern fiind $1 \le \xi \le P$. Funcția cost se poate scrie în forma dată de relația (6.1), unde $1 \le j \le N$ este indicele perceptronilor simpli de ieșire și $e_j(\xi)$ este eroarea obținută în fiecare nod de ieșire. S-a notat cu $e(\xi)$ vectorul erorilor obținute la ieșirile rețelei, pentru fiecare pattern de intrare.

$$C(\mathbf{w}) = \sum_{\xi=1}^{P} C(\mathbf{w}; \mathbf{e}(\xi)) = \sum_{\xi=1}^{P} \sum_{j=1}^{N} C(\mathbf{w}; e_j(\xi)) \tag{6.1}$$

Funcția cost (6.1) se poate scala, prin împărțire la numărul patternurilor și / sau la numărul perceptronilor simpli din stratul de ieșire (relația (5.33)), pentru a face posibilă compararea performanțelor rețelelor cu structură diferită, care au învățat seturi de date cu dimensiuni diferite.

Dacă relația (6.1) se scrie ca (6.2), atunci alegerea funcției cost $C(\mathbf{w})$ devine problema selecției funcției Φ. Sunt posibile două cazuri, în care se cunoaște sau

Scanned with OKEN Scanner

nu densitatea de probabilitate a zgomotului din datele de intrare.

$$C(w) = \sum_{\xi=1}^{P} \sum_{j=1}^{N} \Phi\left(e_j(\xi)\right) \tag{6.2}$$

6.1.1 Alegerea funcţiei cost când funcţia densitate de probabilitate a zgomotului este cunoscută

Dacă se presupune cunoscută funcţia densitate de probabilitate a zgomotului, atunci este cunoscută şi funcţia densitate de probabilitate a erorilor $p_e(e)$ cu e eroare (variabilă aleatoare independentă, cu distribuţie uniformă). Se poate aplica metoda verosimilităţii maxime (ML — "Maximum Likelihood") şi se alege funcţia Φ conform relaţiei (6.3) [C43].

$$\Phi(r) = -\ln p_e(r) \tag{6.3}$$

6.1.1.1 Distribuţie gaussiană a erorilor

Aplicând relaţia (6.3) pentru o distribuţie gaussiană a erorilor (6.4), rezultă că, funcţia cost este o sumă de funcţii $\Phi(r)$ de tip eroare pătratică (6.5). Estimatul optimal al ponderilor pentru problema (3.17) se obţine prin metoda celor mai mici pătrate, când erorile sunt distribuite gaussian [C44].

$$p_e(r) \propto \exp\left(-r^2/2\right) \tag{6.4}$$

$$\Phi(r) \propto r^2 \tag{6.5}$$

6.1.1.2 Distribuţie dublu exponenţială a erorilor

Aplicarea metodei ML pentru distribuţia erorilor dublu exponenţială (6.6), din relaţia (6.3) rezultă funcţia cost de tip normă L_1 a erorii.

$$p_e(r) \propto e^{|r|/2} \tag{6.6}$$

6.1.1.3 Distribuţie uniformă a erorilor

Dacă distribuţia erorilor are tranziţii bine definite şi abrupte, ca în cazul distribuţiei uniforme, atunci este utilă alegerea normei Cebâşev (norma L_∞).

Scanned with OKEN Scanner

6.1.1.4 Distribuţie gaussiană generalizată a erorilor

Burrascano [B4] a propus utilizarea criteriului L_p, cu $1 \le p \le \infty$ arătând că, optimizarea acestuia corespunde ipotezei distribuţiei gaussiene generalizate a erorilor (6.7), unde s-au notat cu μ_e şi σ media şi respectiv, deviaţia standard a distribuţiei; Γ este funcţia gamma, iar indicele p marchează norma L_p. Pentru $p = 2$ se obţine funcţia gaussiană standard.

$$p_e(r) = p^{[1-1/p]} \left[2\,\sigma_p\,\Gamma(1/p)\right]^{-1} \exp\left[-p^{-1}\,|r-\mu_e|^p \Big/ (\sigma_p)^p\right] \tag{6.7}$$

6.1.1.5 Distribuţie Cauchy a erorilor

Liano [L46] a arătat că, pentru distribuţie Cauchy a erorilor, dată de relaţia (6.8), estimatul ML optimal se obţine pentru funcţia cost logaritm din eroarea pătratică medie:

$$p_e(r) = \frac{1}{\pi\left(1+r^2\right)} \tag{6.8}$$

$$\Phi(r) = \log\left(1+\frac{r^2}{2}\right) \tag{6.9}$$

Astfel, dacă se cunoaşte densitatea de probabilitate a erorilor, prin aplicarea metodei ML se pot obţine diferite funcţii cost. Deşi într-un număr mare de aplicaţii ale RNA se alege norma L_2 a erorii (criteriul eroare pătratică minimă) ca funcţie cost, opţiunea este motivată numai de ipoteza că erorile sunt distribuite gaussian, ceea ce nu este totdeauna adevărat. In consecinţă, norma L_2 este optimală numai pentru distribuţie gaussiană a erorilor, estimatul vectorului de ponderi în prezenţa erorilor de tip "outliers"[22] fiind necorespunzător [W29, S42].

6.1.2 Alegerea funcţiei cost când funcţia densitate de probabilitate a zgomotului nu este cunoscută

Dacă funcţia densitate de probabilitate a zgomotului nu este cunoscută, uneori există posibilitatea estimării acesteia. In acest caz, după determinarea formei distribuţiei zgomotului, prin aplicarea testelor statistice sau folosind tot

[22] "Outliers" = date care nu respectă distribuţia generală a datelor. De exemplu, pot fi impulsuri care apar într-o distibuţie normală, etc.

structuri neurale (reţele perceptron multistrat [M79], RBF [H58, N13, M45, M79], etc.), se poate utiliza abordarea prezentată în par. 6.1.1.

Dacă distribuţia zgomotului este dificil de estimat, fiind un amestec de distribuţii sau zgomot dependent de semnal [P16, C43], atunci se poate utiliza funcţia cost (6.2), în care funcţiile Φ sunt alese dintr-o clasă numită "estimatori de tip M" [C44, P17]. O altă posibilitate este utilizarea unor funcţii cost inspirate din teoria informaţiei. Cele două abordări sunt alternative, alegerea acestora fiind dependentă de aplicaţie şi de ipotezele făcute asupra distribuţiei datelor.

6.1.2.1 Utilizarea estimatorilor de tip M

Obiectivul problemei de minimizare (3.17) se poate scrie ca (6.10) sau, echivalent, (6.11), unde s-a notat cu φ derivata de ordinul întâi a funcţiei Φ, care nu se presupune de forma ($-$ *ln* $p(e)$) [C43].

$$\min_{\mathbf{w}} \sum_{\xi=1}^{P} \Phi\big(e(\xi)\big) \tag{6.10}$$

$$\sum_{\xi=1}^{P} \varphi\big(e(\xi)\big) = 0 \tag{6.11}$$

Formularea (6.10) a problemei de minimizare, a fost numită de Cichocki şi Unbehauen "L_2 robustă" ("Robust Least Squares"), datorită înlocuirii funcţiei cost uzuale, eroare pătratică medie, cu funcţii "estimator robust de tip M" [C16]. Criteriul de cost L_2 poate fi considerat un caz special al problemei generale (6.10).

Funcţiile Φ estimatori robuşti sunt cunoscute în statistică şi pot fi alese dintre funcţiile logistică, Huber, Talvar, Hampel, cărora li adaugă normele L_1, L_2, etc. Scopul utilizării lor este reducerea influenţei erorilor de tip impuls sau "outliers" asupra estimării parametrilor **w**. Deşi necesitatea utilizării funcţiilor cost estimatori robuşti a fost subliniată de Cichocki şi Unbehauen, Werbos, ş.a., s-au înregistrat până în prezent puţine utilizări ale acestora în aplicaţii cu RNA [C43, D201, D204, L46].

6.1.2.2 Utilizarea estimatorilor de tip M modificaţi şi variantelor acestora

❒ ***Alegerea unei funcţii cost compuse* (I)**

Karayiannis şi Venetsanopoulos [C16] au propus utilizarea funcţiei cost dată de relaţia (6.12), unde Φ_1, Φ_2 sunt funcţii estimatori robuşti de tip M,

pretutindeni convexe şi diferenţiabile, iar $\theta \in [0, 1]$ este un parametru ce descreşte treptat (de la valoarea 1 la 0) în timpul învăţării. O strategie posibilă de descreştere a parametrului θ este (6.13), unde $c > 0$, iar C este funcţia cost globală, evaluată în sensul normei L_2.

$$C(\mathbf{w}) = (1-\theta)\sum_{\xi=1}^{P}\sum_{j=1}^{N}\Phi_1\left(e_j(\xi)\right) + \theta\sum_{\xi=1}^{P}\sum_{j=1}^{N}\Phi_2\left(e_j(\xi)\right) \tag{6.12}$$

$$\theta(C) = \exp(-c / C^2) \tag{6.13}$$

❒ *Funcţie Φ cu formă adaptivă* (II)

Chen şi Jain au propus [C43] alegerea derivatei φ a funcţiei optimale Φ, de tip tangentă hiperbolică Hampel, cu formă variabilă în timpul învăţării (abscisa maximului funcţiei, precum şi punctele de intersecţie cu axele se modifică în timp) pe baza proprietăţilor statistice ale datelor de intrare. Valoarea maximă a procentajului datelor de tip "outliers" existent în distribuţia dată, este impus.

Funcţia cost propusă este dată de relaţia (6.14), având derivata (6.15). S-au notat cu $a(t)$ şi $b(t)$ abscisele maximului şi respectiv, punctului de anulare a funcţiei (ambele dependente de timp), iar c_1 şi c_2 sunt constante.

$$C = \sum_{\xi=1}^{P}\sum_{j=1}^{N}\Phi\left(e_j(\xi); t\right) \tag{6.14}$$

$$\varphi(r) = \frac{d\Phi(r)}{dr} = \begin{cases} r & \text{pentru } |r| \le a(t) \\ c_1 \tanh\left[c_2\left(b(t) - |r|\right)\right]\operatorname{sgn}(r) & \text{pentru } a(t) < r \le b(t) \\ 0 & |r| > b(t) \end{cases} \tag{6.15}$$

❒ *Funcţia cost Lister* (III)

Lister [G19] a propus funcţia cost EEF ("Exception Error Function"), pentru cazul în care datele prezentate reţelei conţin "excepţii" (date de tip "outliers", zgomot de impulsuri, etc.) şi a arătat performanţele bune ale acesteia pentru probleme de clasificare. Gaynier şi Downs [G19] au modificat această funcţie, obţinând funcţia cost TEF ("Test Error Function") dată de relaţia (6.16), unde e este eroarea la ieşirea reţelei, iar α, β sunt parametri ce se aleg (valorile propuse au fost $\alpha = 1$, $\beta = 1.25, 1, 0.75, 0.5, 0.25, 0.12, 0.06$) [G19].

Scanned with OKEN Scanner

$$C = \frac{\alpha\left(e^2 + e^4\right)}{1 - e^2 + \beta e^4} \tag{6.16}$$

6.1.2.3 Utilizarea funcţiilor ITF ("Information Theoretic Functions")

Atunci când datele de intrare ale modelului neural sunt stohastice, este indicată utilizarea funcţiilor cost ITF [H19], care se construiesc pornind de la definiţiile clasice din teoria informaţiei, ale entropiei, entropiei reunite, entropiei condiţionate şi transinformaţiei [B14, H21, H41, S202, S203].

❒ ***Entropia ca funcţie cost*** **(I)**

Shannon a propus în anul 1948 funcţia entropie $\mathcal{H}$ (incertitudine medie asupra unui set de date cu probabilităţile p_i), definită de relaţia (6.17), unde K este o constantă pozitivă, depinzând de unitatea de măsură:

$$\mathcal{H} = -K\sum_{i=1}^{n} p(x_i)\log p(x_i) \tag{6.17}$$

Entropia ca funcţie cost este maximizată în lucrările [I28, H49, W28]. Kamimura şi Nakanishi [K32] au propus utilizarea ca funcţie cost, a diferenţei dintre entropia maximă şi entropia reală.

❒ ***Funcţia cost Kullback – Liebler*** **(II)**

Pentru două distribuţii cu densităţile de probabilitate p şi q, din relaţia (6.17) se obţine inegalitatea lui Shannon (6.18). Egalitatea are loc numai dacă $p = q$ pentru toate valorile x_i.

$$-\sum_i p(x_i)\log q(x_i) \geq -\sum_i p(x_i)\log p(x_i) \tag{6.18}$$

Pe baza inegalităţii lui Shannon, Kullback a definit în anul 1959 funcţia cost numită divergenţa asimetrică dintre două distribuţii p şi q, "eroarea – G" sau distanţa Kullback – Leibler [B14, H41]:

$$G(p,q) = \sum_i p(x_i)\log\frac{p(x_i)}{q(y_i)} \tag{6.19}$$

Scanned with OKEN Scanner

Pearlmutter şi Hinton [B14] au folosit funcţia cost "eroare – G" pentru cazul simetric ($p = 1 - q$). S-au notat prescurtat cu p, q densităţile de probabilitate ale distribuţiei valorilor obţinute în nodul de ieşire al unei RNA "feedforward".

$$G(p,q) = p\log\left(\frac{p}{q}\right) + (1-p)\log\left(\frac{1-p}{1-q}\right) \tag{6.20}$$

O variantă de aplicare a funcţiei cost Kullback – Liebler pentru o RNA "feedforward" cu mai multe noduri de ieşire ($N > 1$) este propusă de Hertz, Krogh şi Palmer [H19]:

$$C = -\sum_{\xi=1}^{P}\sum_{j=1}^{N}\left[\frac{1}{2}\left(1+d_j(\xi)\right)\log\left(\frac{1+d_j(\xi)}{1+y_j(\xi)}\right) + \frac{1}{2}\left(1-d_j(\xi)\right)\log\left(\frac{1-d_j(\xi)}{1-y_j(\xi)}\right)\right] \tag{6.21}$$

Funcţia cost dată de relaţia (6.21) este strict pozitivă, cu excepţia situaţiei când valorile obţinute la ieşirile reţelei $y_j(\xi)$ sunt egale cu valorile dorite $d_j(\xi)$ (caz în care funcţia se anulează), unde $1 \le j \le N$ şi $1 \le \xi \le P$.

Van Ooyen, ş.a. [H38] au propus pentru aceeaşi structură neurală funcţia cost:

$$C = -\sum_{\xi=1}^{P}\sum_{j=1}^{N}\left[d_j(\xi)\ln\left(y_j(\xi)\right) + \left(1-d_j(\xi)\right)\ln\left(1-y_j(\xi)\right)\right] \tag{6.22}$$

Modificând funcţia cost dată de relaţia (6.22), Humpert a propus utilizarea funcţiei cost compuse (6.12). Expresia funcţiei Φ_1 din primul termen este dată de relaţia (6.23), cu $Q_m(u, a)$ definit de relaţia (6.24), iar coeficienţii c_m de (6.25). Cel de-al doilea termen al funcţiei cost are expresia (6.21).

$$\Phi_j = -\frac{1}{\beta}\sum_{m=1}^{\infty}\left[Q_m\left(u,-(1+t)\right) - Q_m\left(u,+(1-t)\right)\right] \tag{6.23}$$

$$Q_m(u,a) = \left(\frac{1}{c_m^2+a^2}\right)\left[c_m\,\mathrm{arctg}\left(\frac{t-u}{c_m}\right) - a\ln\left|\frac{t-u+a}{\sqrt{c_m^2+a^2}}\right|\right] \tag{6.24}$$

$$c_m = \frac{\pi}{2\beta}(2m-1) \tag{6.25}$$

Scanned with OKEN Scanner

S-a sugerat o strategie de descreştere a parametrului θ, conform relaţiei (6.13), cu C funcţia de cost globală şi $c = 1$ sau, conform unei relaţii locale similare, în care C se înlocuieşte cu cel de-al doilea termen al funcţiei cost, evaluat pentru fiecare nod de ieşire j.

❒ ***Informaţia mutuală (transinformaţia) ca funcţie cost* (III)**

Dacă $p(x_i, y_j)$ este probabilitatea reunită a simbolurilor x_i, y_j atunci entropia distribuţiei reunite (numită entropie reunită) este [B14, H21, H41, S202, S203]:

$$\mathcal{H}(x, y) = -\sum_{i,j} p(x_i, y_j) \log p(x_i, y_j) \tag{6.26}$$

Entropia condiţionată (numită şi echivocaţie) este incertitudinea asupra variabilei x, când se cunoaşte y şi este dată de relaţia (6.27). Informaţia mutuală (transinformaţia) ce se obţine asupra variabilei x când se cunoaşte y este exprimată de (6.28).

$$\mathcal{H}(x \mid y) = -\sum_{i,j} p(x_i, y_j) \log p(x_i \mid y_j) \tag{6.27}$$

$$I_{x;y} = \mathcal{H}(x) + \mathcal{H}(y) - \mathcal{H}(x, y) = \mathcal{H}(x) - \mathcal{H}(x \mid y) = \mathcal{H}(y) - \mathcal{H}(y \mid x) \tag{6.28}$$

Informaţia mutuală dintre stratul de intrare şi cel ascuns al unei structuri "feedforward" de tip perceptron multistrat, este minimizată de Deco şi Martinetz [D31], Harvey, ş.a. [H48]. Pe de altă parte, transinformaţia este maximizată de Brause [B24] şi se poate arăta că, această abordare este echivalentă minimizării entropiei condiţionate.

Linsker a formulat în anul 1987 principiul conservării informaţiei maxime pentru RNA, principiu independent de regula de învăţare utilizată pentru implementarea sa [H41].

6.1.2.5 Alte funcţii de cost

Deseori, relaţiile de modificare a ponderilor în procesul de învăţare sunt formulate explicit, dar nu şi expresia funcţiei cost. In acest caz, se realizează o formulare indirectă a criteriului de cost.

Există funcţii cost, utilizate mai ales în învăţarea nesupervizată, bazate pe evaluarea distanţelor (distanţa euclidiană dintre vectorul ponderilor şi vectorul datelor de intrare, în cazul structurii Kohonen, etc.) [H7, R25], pe ortogonalitate, corelaţie, covariaţia valorilor de ieşire, etc.

Scanned with OKEN Scanner

6.2 ALEGEREA FUNCŢIEI COST PENTRU O APLICAŢIE DATĂ

Am precizat faptul că, alegerea funcţiei cost depinde de aplicaţia dată: clasificare, estimare / filtrare, etc. Cu excepţia autorilor Niu şi Fischer [N17], care au precizat variante de alegere pentru diferite probleme, majoritatea studiilor efectuate se referă la aplicaţii de clasificare.

O RNA "feedforward" cu M şi N perceptroni simpli în straturile de intrare, respectiv de ieşire, funcţionează ca un clasificator, dacă atribuie vectorul aleator x prezentat la intrările reţelei, uneia dintre cele N clase de ieşire, care are probabilitatea aposteriori maximă [N20]. Hampshire II şi Pearlmutter [H50] au prezentat argumente în favoarea echivalenţei dintre clasificatorul neural "feedforward" perceptron multistrat şi clasificatorul Bayes, pentru anumite funcţii cost, printre care şi funcţia eroare pătratică medie. Afirmaţia nu este însă general valabilă, în sensul că RPM nu sunt doar estimatori Bayes mai deosebiţi, ci permit împărţirea funcţiilor cost utilizate în probleme de clasificare în două categorii: funcţii rezonabile şi funcţii nerezonabile.

6.2.1 Funcţii rezonabile

Funcţiile cost rezonabile sunt cele pentru care se obţin performanţe similare celor ale clasificatorului Bayes, adică pentru care reţeaua neurală aproximează asimptotic probabilităţile Bayes aposteriori ale claselor din care fac parte datele setului de învăţare [T16, L37, S45, N18]. Eroarea pătratică face parte din această clasă, precum şi câteva dintre funcţiile ITF.

Funcţiile rezonabile sunt date de relaţia (6.29), cu y valoarea ieşirii. Pentru $r = 0$ se obţin funcţiile cost ITF. Relaţia (6.30) stă la baza construirii funcţiilor cost: entropie, funcţia Kullback – Liebler, informaţie mutuală [H50, U3]. Pentru $r = 1$ se obţine funcţia cost L_2 dată de relaţia (6.31).

$$C(y) = \int y^r (1-y)^{r-1}\, dy \tag{6.29}$$

$$C(y) = \int (1-y)^{-1}\, dy = -\log(1-y) \tag{6.30}$$

$$C(y) = \int y\, dy = \frac{1}{2} y^2 \tag{6.31}$$

6.2.2 Funcţii care nu sunt rezonabile

Din categoria funcţiilor care nu sunt rezonabile fac parte funcţiile ce aproximează performanţele obţinute de funcţia discriminant Bayes, dar nu obţin la ieşirea reţelei valori care să reflecte probabilităţile aposteriori ale claselor din care fac parte datele de intrare. Un exemplu de funcţii nerezonabile sunt funcţiile cost CFM ("Classification Figures of Merit"), funcţia cost Minkowski şi funcţiile care admit valori dorite nebinare la ieşirile reţelei [H50].

Este important de remarcat faptul că, alegerea unei funcţii care nu este rezonabilă este posibilă pentru o aplicaţie de clasificare dată şi poate fi argumentată, dacă estimatul obţinut pentru parametrii structurii neurale nu are deplasament ("bias") semnificativ.

6.2.2.1 Funcţiile cost CFM ("Classification Figures of Merit")

Funcţiile cost CFM ("Classification Figures of Merit") au fost introduse de Hampshire şi Waibel [B8]. Pentru o problemă de clasificare cu mai multe clase, se presupune că, un perceptron simplu de ieşire este atribuit fiecărei clase (numărul claselor este egal cu numărul nodurilor de ieşire, N). Funcţia cost CFM urmăreşte maximizarea diferenţei dintre valoarea de ieşire a nodului ce reprezintă clasa căreia îi aparţine patternul de intrare şi ieşirea oricărui alt nod de ieşire.

Valoarea funcţiei CFM pentru un pattern de intrare ξ, care aparţine clasei C_n, cu $1 \le n \le N$, este dată de relaţia:

$$C = \sum_{j=1;\ j\neq n}^{N} \frac{1}{1+\exp\left[-\beta\left(y_n - y_j\right)\right]} \tag{6.32}$$

S-au notat cu y_j, y_n valorile de ieşire ale nodurilor j şi n, iar β este un parametru al procesului de învăţare. Când valoarea medie a funcţiei cost C devine maximă, ieşirea corectă y_n tinde să aibă o valoare mare, iar toate celelalte ieşiri tind către valori neglijabile [B8].

6.2.2.2 Funcţia cost Minkowski (norma L_R)

Funcţia cost Minkowski (norma L_R) este definită de relaţia (6.33). Pentru $R = 2$, devine funcţia cost (rezonabilă) L_2 [H50].

$$C(y) = y^R \tag{6.33}$$

Scanned with OKEN Scanner

6.2.2.3 Funcţia cost cu valori nebinare ale ieşirilor dorite

Toate funcţiile cost cu valori nebinare ale ieşirilor dorite sunt considerate nerezonabile pentru probleme de clasificare [H50].

6.2.3 Alte funcţii cost

Telfer şi Szu [T16] au definit pentru RNA "feedforward" perceptron multistrat un criteriu de cost care conduce la aproximarea graniţelor claselor. Se presupune că există numai două clase. Funcţia cost în varianta L_1 este dată de relaţia (6.34) şi aparţine funcţiilor rezonabile din prima categorie, iar cea în varianta L_2 este (6.35) şi aparţine funcţiile rezonabile de tip Minkowski, cu $R = 2$. Funcţia de activare f este funcţia treaptă unipolară (3.8).

$$C = P - \sum_{\xi=1}^{P} f\left[\left(2\,d(\xi) - 1\right) \mathbf{w}^T \mathbf{x}(\xi)\right] = \sum_{\xi=1}^{P} \left\{1 - f\left[\left(2\,d(\xi) - 1\right) \mathbf{w}^T \mathbf{x}(\xi)\right]\right\} \quad (6.34)$$

$$C = \sum_{\xi=1}^{P} \left[d(\xi) - f\left(\mathbf{w}^T \mathbf{x}(\xi)\right)\right]^2 \quad (6.35)$$

6.3 PROIECTAREA REŢELELOR "FEEDFORWARD" CU FUNCŢII COST ESTIMATORI ROBUŞTI DE TIP M

In rezolvarea problemelor de clasificare au fost utilizate majoritatea funcţiilor ITF [H50, L37, N18, S45, T16, U3, ş.a.] şi au fost propuse multe variante ale acestora.

Pe de altă parte, există numai câteva aplicaţii ale funcţiilor cost estimatori robuşti în studiul RNA [C43, D201, D204, L46], deşi acestea sunt bine cunoscute în statistica matematică, iar necesitatea alegerii altor funcţii cost decât norma L_2 a fost argumentată [C16, W29, ş.a.]. Printre motivele utilizării frecvente a normei L_2, chiar în aplicaţii în care acest lucru nu este justificat, se numără:

❶ Norma L_2 (euclidiană) a erorii poate fi aproximată în scopul implementării rapide.

Scanned with OKEN Scanner

❷ Dintre funcţiile estimatori robuşti, funcţia L_1 nu este diferenţiabilă. O soluţie este aproximarea sa printr-o funcţie diferenţiabilă, de exemplu funcţia logistică, dar se adaugă un termen suplimentar de eroare.

❸ O parte dintre funcţiile estimatori robuşti depind de un parametru de control $\beta > 0$, dependent de aplicaţie. Alegerea sa este de obicei empirică.

❹ Uneori, alegerea funcţiei cost estimator robust nu este simplă, dacă distribuţia zgomotului este un amestec de distribuţii.

In continuare, va fi abordată proiectarea RNA "feedforward" cu funcţii cost estimatori robuşti, formulată ca problemă de optimizare în variantele mono – criteriu şi multi – criteriu [D205].

6.3.1 Problema de proiectare ca optimizare mono – criteriu

6.3.1.1 Formularea problemei de optimizare mono – criteriu

Pentru formularea problemei de proiectare a RNA "feedforward" în varianta mono – criteriu, se utilizează (3.17) şi relaţiile (3.23), (6.1), (6.10), şi rezultă:

Minimizează funcţia cost scalară

$$C(\mathbf{w}) = \sum_{\xi=1}^{P} \sum_{j=1}^{N} C\left(\mathbf{w}; e_j(\xi)\right) = \sum_{\xi=1}^{P} \sum_{j=1}^{N} \Phi\left(e_j(\xi)\right) \qquad (6.36)$$

cu constrângerile $\mathbf{w} \in \Re^{\Pi}$

S-a notat cu e_j eroarea în fiecare nod j de ieşire al reţelei $1 \le j \le N$, calculată ca diferenţa dintre valoarea dorită d_j şi cea obţinută y_j.

Funcţiile Φ sunt estimatori robuşti de tip M (tab. 6.1, unde se înlocuieşte variabila r cu $e_j(\xi)$), iar $1 \le \xi \le P$ este indicele patternului de intrare în reţea. Utilizarea funcţiilor Φ reduce influenţa erorilor de tip impuls sau "outliers" asupra estimării parametrilor $\mathbf{w}$, prin constrângerea erorii instantanee e să aibă valori mai mici decât parametrul $\beta > 0$.

Funcţia cost $C(\mathbf{w})$ satisface teorema 3.1.

Scanned with OKEN Scanner

Tab. 6.1 Funcţii $\Phi(r)$ estimatori robuşti de tip M

Funcţii $\Phi(r)$	$\lvert r\rvert \le \beta$	$\lvert r\rvert > \beta$
Norma L_1	$\Phi(r) = \lvert r\rvert$	$\Phi(r) = \lvert r\rvert$
Norma L_2	$\Phi(r) = \frac{1}{2} r^2$	$\Phi(r) = \frac{1}{2} r^2$
Funcţia Huber	$\Phi(r) = \frac{1}{2} r^2$	$\Phi(r) = \beta \lvert r\rvert - \frac{1}{2}\beta^2$
Funcţia Hampel	$\Phi(r) = \frac{\beta^2}{\pi}\left(1 - \cos\frac{\pi r}{\beta}\right)$	$\Phi(r) = 2\frac{\beta^2}{\pi}$
Funcţia logistică	$\Phi(r) = \beta^2 \ln\left[\cosh\left(\frac{r}{\beta}\right)\right]$	$\Phi(r) = \beta^2 \ln\left[\cosh\left(\frac{r}{\beta}\right)\right]$

6.3.1.2 Modificarea valorilor parametrilor în algoritmul BKP, varianta standard

Se consideră o structură neurală "feedforward" perceptron multistrat şi notaţiile din par. 3.2.1. Funcţia de activare a perceptronului simplu este tangenta sigmoidală bipolară (3.9), având panta s. Pentru stratul ascuns, valorile net_h la intrarea în stratul ascuns sunt calculate cu relaţia (3.2).

Prin comparaţia valorilor de ieşire $y_j = f(net_h)$ obţinute din (3.6), cu cele dorite d_j, pentru fiecare pattern de intrare ξ, cu $1 \le \xi \le P$, se poate determina eroarea $e_j(\xi) = d_j(\xi) - y_j(\xi)$. In algoritmul BKP standard [R3, ş.a.] se utilizează funcţia cost (eroare) normă L_2 a erorii.

Valoarea totală a funcţiei cost, pentru toate patternurile de intrare $x(\xi)$, $1 \le \xi \le P$, se determină cu relaţia (6.1).

Ponderile sunt modificate prin propagarea inversă a erorii (relaţia (5.34), cu $\alpha = 0$), în scopul determinării vectorului de ponderi optim, care minimizează funcţia cost. Condiţia impusă este anularea gradientului funcţiei cost C în raport cu vectorul $\mathbf{w}$. Calculul gradientului conduce la relaţiile (6.37) şi (6.38) pentru modificarea ponderilor conexiunilor dintre stratul de intrare şi cel ascuns, respectiv a celor dintre stratul ascuns şi cel de ieşire, utilizând expresiile (6.39) şi (6.40) [D205].

$$v_{hj}(k+1) - v_{hj}(k) = -\eta \frac{\partial C(\xi)}{\partial v_{hj}} = -\eta \frac{\partial C(\xi)}{\partial y_j(\xi)} \frac{\partial y_j(\xi)}{\partial net_j(\xi)} \frac{\partial net_j(\xi)}{\partial v_{hj}} = \eta \delta_j z_h \tag{6.37}$$

$$w_{ih}(k+1)-w_{ih}(k)=-\eta\frac{\partial C(\xi)}{\partial w_{ih}}=-\eta\frac{\partial C(\xi)}{\partial z_h(\xi)}\frac{\partial z_h(\xi)}{\partial net_h(\xi)}\frac{\partial net_h(\xi)}{\partial w_{ih}}=$$
$$=\eta\delta_h x_i \qquad (6.38)$$

$$\delta_j(\xi)=-\frac{\partial C(\xi)}{\partial y_j(\xi)}\frac{\partial y_j(\xi)}{\partial net_j(\xi)}=-\frac{\partial C(\xi)}{\partial y_j(\xi)}f_j'\left(net_j(\xi)\right)=\ f_j'\left(net_j(\xi)\right)\frac{\partial C\left(e_j(\xi)\right)}{\partial y_j(\xi)}$$

(6.39)

$$\delta_h(\xi)=f_h'\left(net_h(\xi)\right)\sum_m \delta_m(\xi)\ w_{hm} \qquad (6.40)$$

S-au utilizat notaţiile:

δ_j	eroarea în nodul de ieşire *j*, calculată cu relaţia (6.39)
δ_h	eroarea în nodul ascuns *h*, calculată cu relaţia (6.40)
m	indice care se referă la toţi perceptronii simpli din stratul precedent (din care se propagă eroarea).
ξ	indexul patternului de intrare $\boldsymbol{x}$, $1 \le \xi \le P$
f ′	derivata de ordinul întâi a funcţiei de activare a nodului
z_h	valoarea de ieşire din nodul ascuns h, unde $1 \le h \le H$

Printre soluţiile de creştere a vitezei de convergenţă şi de evitare a blocării algoritmului în puncte de minim local, se numără utilizarea parametrilor adaptivi (prag, moment, panta funcţiei de activare). Varianta de modificare a ponderilor cu termen moment $0 < \alpha < 1$ este (5.34), care conduce la relaţiile (6.41) pentru modificarea ponderilor conexiunilor dintre stratul de intrare şi cel ascuns, respectiv (6.42) pentru cele dintre stratul ascuns şi stratul de ieşire:

$$w_{ih}(k+1)-w_{ih}(k)=\eta\ \delta_h\ x_i+\alpha\ [w_{ih}(k)-w_{ih}(k-1)] \qquad (6.41)$$

$$v_{hj}(k+1)-v_{hj}(k)=\eta\ \delta_j\ z_h+\alpha\ [v_{hj}(k)-v_{hj}(k-1)] \qquad (6.42)$$

Valoarea parametrului α poate fi stabilită empiric, sau poate fi modificată adaptiv. Relaţii similare se pot scrie pentru modificarea pragurilor, dacă fiecare dintre acestea se consideră o pondere suplimentară pentru un perceptron simplu din strat, care are totdeauna valoarea semnalului de intrare egală cu +1.

Scanned with OKEN Scanner

6.3.1.3 Modificarea valorilor parametrilor în algoritmul BKP, pentru funcţii cost estimatori robuşti de tip M

Alegerea funcţiilor cost diferite de norma L_2 a erorii (tab. 6.1) implică rescrierea relaţiilor matematice (6.37) – (6.40). Se poate observa că, în expresia generală de modificare a ponderilor (5.34), respectiv în expresiile (6.37) şi (6.38) pentru fiecare set de ponderi, se modifică numai relaţia de calcul pentru eroarea elementară în fiecare nod de ieşire (6.39), conform tab. 6.2 [D201, D204, L205]. Parametrul β apare în expresia funcţiilor cost Huber, Hampel şi logistică.

Tab. 6.2 Relaţia de calcul pentru $\delta_j(\xi)$, depinzând de funcţia cost aleasă

Funcţia cost	$\delta_j(\xi)$	
Norma L_1	$-\operatorname{sgn}\left(e_j(\xi)\right) f_j'\left(net_j(\xi)\right)$	(6.43)
Norma L_2	$-f_j'\left(net_j(\xi)\right)\left(d_j(\xi)-y_j(\xi)\right)$	(6.44)
Funcţia Huber	$\begin{cases} -f_j'\left(net_j(\xi)\right)\left(d_j(\xi)-y_j(\xi)\right) & , \lvert e_j(\xi)\rvert \le \beta \\ -\beta f_j'\left(net_j(\xi)\right)\operatorname{sgn}\left(e_j(\xi)\right) & , \lvert e_j(\xi)\rvert > \beta \end{cases}$	(6.45)
Funcţia Hampel	$\begin{cases} -\frac{\beta\pi}{2} f_j'\left(net_j(\xi)\right)\sin\left(\frac{\pi e_j(\xi)}{\beta}\right) & , \lvert e_j(\xi)\rvert \le \beta \\ 0 & , \lvert e_j(\xi)\rvert > \beta \end{cases}$	(6.46)
Funcţia logistică	$-\beta f_j'\left(net_j(\xi)\right)\tanh\left(\frac{e_j(\xi)}{\beta}\right)\cosh^{-1}\left(\frac{e_j(\xi)}{\beta}\right)$	(6.47)

6.3.1.4 Modificarea valorilor parametrilor în algoritmul BKP, pentru funcţii cost estimatori robuşti de tip M şi pantă variabilă a funcţiei de activare a perceptronului simplu

Panta funcţiei de activare a nodului (perceptron simplu) este de obicei menţinută constantă în timpul învăţării. S-a arătat că, modificarea acesteia în timpul procesului de învăţare, ca şi modificarea ponderilor şi pragurilor, conduce la îmbunătăţirea vitezei de convergenţă.

In continuare, rescriem relaţiile (6.37) – (6.40), pentru cazul pantei variabile a funcţiei de activare [D201, D204].

Scanned with OKEN Scanner

Fie funcţia de activare (3.9), considerată ca funcţie de două argumente *f(s, net)*. Pentru stratul de ieşire se poate scrie relaţia (6.48), iar pentru cel ascuns relaţia (6.49), care conduc la expresia (6.50) a modificării pantei funcţiei de activare a oricărui perceptron simplu. S-a notat cu χ rata de modificare / învăţare a pantei funcţiei de activare.

$$\frac{\partial C(\xi)}{\partial s} = \frac{\partial C(\xi)}{\partial y_j(\xi)} \frac{\partial y_j(\xi)}{\partial z_h(\xi)} \frac{\partial z_h(\xi)}{\partial s} = -\delta_j(\xi) \left(\frac{\partial f(s, net)}{\partial net}\right)^{-1} \left(\frac{\partial f(s, net)}{\partial s}\right) \tag{6.48}$$

$$\frac{\partial C(\xi)}{\partial s} = \frac{\partial C(\xi)}{\partial z_h(\xi)} \frac{\partial z_h(\xi)}{\partial x_i(\xi)} \frac{\partial x_i(\xi)}{\partial s} = -\delta_h(\xi) \left(\frac{\partial f(s, net)}{\partial net}\right)^{-1} \left(\frac{\partial f(s, net)}{\partial s}\right) \tag{6.49}$$

$$s(k+1) - s(k) = \chi\left(-\frac{\partial C(\xi)}{\partial s}\right) = \chi\,\delta \left(\frac{\partial f(s, net)}{\partial s}\right)\left(\frac{\partial f(s, net)}{\partial net}\right)^{-1} \tag{6.50}$$

Valoarea δ este calculată cu una dintre expresiile (6.43) – (6.47) pentru nodurile de ieşire şi cu (6.40) pentru nodurile stratului ascuns. Se poate introduce un termen moment $0 < \rho < 1$ în expresia (6.50), similar celui din relaţia de modificare a ponderilor:

$$s(k+1) - s(k) = \chi\left(-\frac{\partial C(\xi)}{\partial s}\right) + \rho\,(s(k) - s(k-1)) \tag{6.51}$$

6.3.2 Problema de proiectare ca optimizare multi – criteriu

Alegerea estimatorilor robuşti de tip M este dificilă, dacă distribuţia zgomotului este un amestec de distribuţii, deseori altele decât cele standard. O soluţie posibilă este utilizarea estimatorilor de tip M modificaţi sau a variantelor acestora (par. 6.1.2). Se urmăreşte minimizarea simultană a mai multor funcţii, ceea ce impune reformularea problemei de optimizare (6.36). Aceasta devine o problemă de optimizare multi – criteriu (numită şi multi – performanţă sau multi – obiectiv [C16]).

Scanned with OKEN Scanner

6.3.2.1 Formularea problemei de optimizare multi – criteriu

Problema de proiectare a RNA "feedforward" în varianta de optimizare multi – criteriu, pentru funcţii cost estimatori robuşti de tip M este [D205]:

Minimizează funcţia cost vectorială $\underline{C}(\mathbf{w}) = [C_1(\mathbf{w})\ C_2(\mathbf{w})]^T$

cu constrângerile $\mathbf{w} \in \Re^{\Pi}$ (6.52)

unde funcţiile cost $C_1(\mathbf{w})$, $C_2(\mathbf{w})$ sunt date de relaţiile (6.53), (6.54), iar $\Phi_1(e_j(\xi))$, $\Phi_2(e_j(\xi))$ sunt estimatori robuşti de tip M (tab. 6.1).

$$C_1(\mathbf{w}) = \sum_{\xi=1}^{P} \sum_{j=1}^{N} \Phi_1\left(e_j(\xi)\right) \tag{6.53}$$

$$C_2(\mathbf{w}) = \sum_{\xi=1}^{P} \sum_{j=1}^{N} \Phi_2\left(e_j(\xi)\right) \tag{6.54}$$

Am considerat că, funcţia vectorială $\underline{C}(\mathbf{w})$ are numai două componente, deşi numărul acestora poate fi mai mare. Soluţia minimizării simultane a funcţiilor $C_1(\mathbf{w})$, $C_2(\mathbf{w})$ reprezintă un compromis. Intrucât nu există o soluţie optimală $\mathbf{w}^*$, care să satisfacă relaţiile $C_1(\mathbf{w}^*) \le C_1(\mathbf{w})$ şi $C_2(\mathbf{w}^*) \le C_2(\mathbf{w})$, pentru toţi vectorii $\mathbf{w}$ din spaţiul ponderilor, se determină o soluţie optimală Pareto (pentru aceasta, îmbunătăţirea unui criteriu are drept efect înrăutăţirea celui de-al doilea criteriu).

Pentru a rezolva problema de optimizare multi – criteriu (6.52), aceasta trebuie transformată într-o problemă de optimizare scalară echivalentă, cu o singură funcţie cost. Dacă se aplică metoda normei L_p [S52] şi se presupune că, funcţiile $C_1(\mathbf{w}) \ge 0$, $C_2(\mathbf{w}) \ge 0$, pentru toţi vectorii $\mathbf{w}$ din spaţiul ponderilor, atunci problema de optimizare (6.52) se poate rescrie sub forma (6.55). Pentru $p = 1$, aceasta devine (6.56), cu funcţia cost scalară obţinută prin ponderarea componentelor $C_1(\mathbf{w})$, $C_2(\mathbf{w})$. In cazul particular $\theta_1 = 1 - \theta$, $\theta_2 = \theta$, funcţia cost scalară din problema (6.56) devine (6.12).

O altă posibilitate de rezolvare a problemei de optimizare multi – criteriu (6.52), este aplicarea metodei cu constrângeri ε, care se bazează pe minimizarea unei singure funcţii cost, de exemplu $C_1(\mathbf{w})$, funcţia $C_2(\mathbf{w})$, fiind considerată drept constrângere. Problema de optimizare se poate scrie sub forma (6.57).

Scanned with OKEN Scanner

Minimizează funcţia cost scalară

$$C(\mathbf{w}) = \left\{ [\theta_1 \, C_1(\mathbf{w})]^p + [\theta_2 \, C_2(\mathbf{w})]^p \right\}^{1/p}, \qquad \text{cu } 1 \le p \le \infty$$

cu constrângerile: (6.55)

$\mathbf{w} \in \Re^{\Pi}$

$\theta_1 + \theta_2 = 1$

$\theta_1 \ge 0, \ \theta_2 \ge 0$

Minimizează funcţia cost scalară

$$C(\mathbf{w}) = \theta_1 \, C_1(\mathbf{w}) + \theta_2 \, C_2(\mathbf{w})$$

cu constrângerile: (6.56)

$\mathbf{w} \in \Re^{\Pi}$

$\theta_1 + \theta_2 = 1$

$\theta_1 \ge 0, \ \theta_2 \ge 0$

Minimizează funcţia cost scalară $C_1(\mathbf{w})$

cu constrângerile: (6.57)

$\mathbf{w} \in \Re^{\Pi}$

$C_2(\mathbf{w}) \le \varepsilon$

Problema de optimizare sub forma (6.56) prezintă constrângeri atât de tip egalitate, cât şi de inegalitate, iar sub forma (6.57) conţine numai constrângeri de inegalitate, fiind o particularizare a formei generale (3.25). In cazul problemei (6.57), presupunând $\mathbf{w} = [w_1, \ldots, w_{\Pi}]^T$, funcţia Lagrange (3.26) devine:

Scanned with OKEN Scanner

$$L(\mathbf{w},\lambda) = C_1(\mathbf{w}) + \lambda\,[\varepsilon - C_2(\mathbf{w})] \tag{6.58}$$

unde λ este multiplicator Lagrange. Presupunând că există o soluţie a problemei, notată cu $\mathbf{w}^* = [w_1^*, w_2^*, \ldots, w_\Pi^*]$, pentru care $\min_{\mathbf{w}} C_1(\mathbf{w}) = C_1(\mathbf{w}^*)$, astfel încât constrângerea $[\varepsilon - C_2(\mathbf{w}^*)] \geq 0$ este satisfăcută, atunci sunt adevărate condiţiile Kuhn – Tucker (3.27) – (3.30), care se pot scrie:

$$\frac{\partial C_1(\mathbf{w}^*)}{\partial w_q} - \lambda^* \frac{\partial C_2(\mathbf{w}^*)}{\partial w_1} = 0, \quad 1 \leq q \leq \Pi \tag{6.59}$$

$$\varepsilon - C_2(\mathbf{w}^*) \geq 0 \tag{6.60}$$

$$\lambda^* \leq 0 \tag{6.61}$$

$$\lambda^* [\varepsilon - C_2(\mathbf{w}^*)] = 0 \tag{6.62}$$

Pentru rezolvarea setului de ecuaţii se poate utiliza metoda gradientului negativ.

6.3.2.2 Modificarea valorilor parametrilor în algoritmul BKP, pentru funcţii cost estimatori robuşti de tip M modificaţi

Se consideră cazul particular $\theta_1 = 1 - \theta$, $\theta_2 = \theta$ în problema de optimizare (6.56). Funcţia cost scalară ce trebuie minimizată este dată de relaţia (6.2), unde:

$$\Phi(e_j(\xi)) = (1-\theta)\,\Phi_1(e_j(\xi)) + \theta\,\Phi_2(e_j(\xi)) \tag{6.63}$$

Se aleg funcţiile estimatori robuşti (tab. 6.1) astfel: funcţia $\Phi_1(e_j(\xi))$ este normă L_2 a erorii, iar $\Phi_2(e_j(\xi))$ este funcţia logistică.

Relaţiile de modificare a ponderilor în algoritmul BKP sunt (6.37) şi (6.38) pentru ponderile conexiunilor dintre stratul ascuns şi stratul de ieşire, respectiv dintre stratul de intrare şi stratul ascuns. Erorile elementare în nodurile ascunse sunt date de relaţia (6.40), iar în nodurile de ieşire de (6.64), în care erorile elementare corespunzătoare funcţiei cost L_2 şi funcţiei logistice au fost prezentate în tab. 6.2.

Scanned with OKEN Scanner

$$\delta_j(\xi) = (1-\theta)\,\delta_j(\xi)_{\text{pentru } L_2} + \theta\,\delta_j(\xi)_{\text{pentru Log.}} \tag{6.64}$$

Pentru a stabili strategia de modificare a parametrului θ la fiecare epocă, se evaluează influenţa erorilor de tip "ouliers" asupra valorii funcţiei $\Phi(e_j(\xi))$ (deci şi asupra funcţiei cost globale), calculând:

$$\lim_{r\to\infty} \frac{\partial \Phi(r)}{\partial r} = \lim_{r\to\infty} \frac{\partial}{\partial r}\left[(1-\theta)\,\Phi_1(r) + \theta\,\Phi_2(r)\right] \tag{6.65}$$

Influenţa erorilor "outliers" este minimă dacă limita este nulă sau, cel puţin finită, cu valoare mică. Inlocuind expresiile funcţiilor estimatori robuşti în relaţia (6.65), din calcul rezultă condiţiile pe care trebuie să le îndeplinească parametrul θ în timpul învăţării. Astfel, se poate arăta că, strategia de modificare dată de expresia (6.13) este corectă, dar impune alegerea empirică a constantei $c > 0$. Pentru a evita acest lucru, s-a propus modificarea parametrului θ la fiecare epocă, conform relaţiei [D205]:

$$\theta(C) = \exp\left(-\frac{\mu_{x\ test}}{C^2}\right) \tag{6.66}$$

unde $\mu_{x\ test}$ este media distribuţiei datelor de test, iar C este eroarea totală la ieşire acumulată până în epoca curentă. Din relaţia (6.63) se observă că, la începutul învăţării, eroarea are valori mari, deci $\theta \to 1$ şi termenul L_2 nu are nici o contribuţie în valoarea funcţiei cost globale. Când eroarea scade, $\theta \to 0$ şi contribuţia termenului logistic se diminuează.

6.3.2.3 Modificarea valorilor parametrilor în algoritmul BKP, pentru funcţii cost estimatori robuşti de tip M modificaţi şi pantă variabilă a funcţiei de activare a perceptronului simplu

Fie condiţiile impuse la punctul 6.3.2.2. Dacă panta funcţiei de activare a perceptronului simplu este variabilă, atunci relaţiilor de modificare a ponderilor (6.37) şi (6.38), li se adaugă relaţia de modificare a pantei (6.50). In expresiile (6.48) şi (6.49) ale derivatelor funcţiei cost în raport cu pantele funcţiilor de activare ale nodurilor de ieşire, respectiv ale nodurilor ascunse, erorile elementare sunt date de relaţia (6.40) pentru nodurile ascunse şi de (6.64) pentru nodurile de ieşire, cu termenii corespunzători funcţiei cost L_2 şi funcţiei logistice din tab. 6.2.

Scanned with OKEN Scanner

6.4 REZULTATE EXPERIMENTALE ALE PROIECTĂRII CU FUNCŢII COST ESTIMATORI ROBUŞTI DE TIP M

6.4.1 Problema de studiu

Din punct de vedere teoretic, problema de studiu abordată este estimarea parametrilor. Dacă distribuţia datelor este cunoscută (respectiv, modelul care a generat datele), atunci aceasta poate fi realizată ca o estimare parametrică în variantă "clasică" [S202, S203, P17, H21]. Intrucât în majoritatea cazurilor reale, distribuţia datelor nu este cunoscută sau, nu se identifică nici uneia dintre distribuţiile clasice (uniformă, gaussiană, exponenţială, Cauchy, Laplace, etc.), datorită existenţei aşa numitelor date "outliers", se impune utilizarea abordărilor neparametrice.

Problema studiată în continuare, pentru verificarea experimentală a precizărilor teoretice din par. 6.3, se referă la estimarea potenţialelor evocate, folosind o reţea neurală "feedforward" perceptron multistrat.

In cercetarea activităţii electrice a creierului, se studiază atât activităţile spontane, cât şi cele evocate. Un instrument important este electroencefalograma (EEG), o înregistrare a activităţii electrice a creierului. Activitatea EEG se numeşte spontană [W10] dacă nu există nici o intervenţie exterioară vizibilă care să determine modificare sa, respectiv evocată, dacă este influenţată de stimuli centrali sau periferici.

Definiţia 6.1 Potenţialul evocat (PE) este răspunsul la un stimul.

Acesta poate fi pus în evidenţă utilizând microelectrozi, macroelectrozi plasaţi pe cortex sau electrozi plasaţi pe craniu. In ultimul caz este necesară extragerea PE din semnalul EEG, care îl maschează [W10]. Problema este des întâlnită în prelucrarea semnalelor medicale, constând în mascarea unui semnal fiziologic de către semnale cu valori mult mai mari.

Semnalul PE, care trebuie estimat, este un semnal de amplitudine mică în comparaţie cu semnalul EEG, care poate fi considerat drept fundal de zgomot; de asemenea, EEG este un semnal neliniar. Ipotezele iniţiale sunt: ① zgomotul (semnal EEG) este aditiv; ② semnalele EEG şi PE nu sunt corelate. Atunci se poate scrie relaţia (6.67), în care s-au notat cu $x_{PE\ corect}(t)$ potenţialul evocat corect (fără zgomot) şi cu $x_{PE\ cu\ zgomot}(t)$ semnalul măsurat; $n(t)$ este activitatea EEG.

$$x_{PE\ cu\ zgomot}(t) = x_{PE\ corect}(t) + n(t) \qquad (6.67)$$

Scanned with OKEN Scanner

Se doreşte aflarea unui estimat $\hat{x}(t)$ astfel încât funcţia cost $C(\mathbf{w})$ (3.23) să fie minimizată, unde $\mathbf{w}$ sunt parametrii modelului neural. S-a notat cu $C(\mathbf{w}; x)$ norma (6.68), în sens L_2.

$$C(\mathbf{w}; \mathbf{x}) = \| \mathbf{x}_{PE\ corect}(t) - \hat{\mathbf{x}}(t) \| \qquad (6.68)$$

Vom considera studiul în timp discret, de aceea în continuare se va utiliza indicele k în loc de t.

6.4.1.1 Setul datelor

Semnalul potenţial evocat corect a fost generat sub forma unei sinusoide complete, urmată de o jumătate de sinusoidă atenuată. Zgomotul de impulsuri n_i (6.69) a fost obţinut prin filtrarea unei secvenţe de numere aleatoare q_i, cu distribuţie uniformă, cu ajutorul unui filtru cu 11 coeficienţi [U4]. Potenţialul evocat cu zgomot se obţine din relaţia (6.67).

$$n_i = (-36\, q_{i-5} + 9\, q_{i-4} + 44\, q_{i-3} + 69\, q_{i-2} + 84\, q_{i-1} + 89\, q_i + 84 + q_{i+1} + 69\, q_{i+2} + 44\, q_{i+3} + 9\, q_{i+4} - 36\, q_{i+5}) / 429 \qquad (6.69)$$

Zgomotul nefiltrat (200 eşantioane) şi zgomotul filtrat (200 eşantioane) sunt reprezentate în fig. 6.1, respectiv 6.2. Fig. 6.3 şi 6.4 conţin graficele potenţialului evocat corect şi celui zgomotos, pentru 200 şi, respectiv, 576 eşantioane.

Seturile de date prezentate reţelei au fost construite citind cele P eşantioane ale semnalul generat anterior, printr-o fereastră de dimensiune W eşantioane (W impar). Fiecare pattern a fost constituit din W eşantioane, citite pe măsură ce fereastra s-a deplasat cu câte un pas (eşantion) spre dreapta. A rezultat un set de P patternuri de intrare. Valoarea de ieşire dorită a fost cea din centrul ferestrei. Au rezultat matrici de date corecte, respectiv cu zgomot, fiecare de dimensiuni W x P, precum şi un vector al valorilor de ieşire dorite, de dimensiune 1 x P.

Valorile numerice pentru W şi P sunt precizate în paragraful următor.

6.4.1.2 Modelul neural

❒ ***Structura reţelei neurale***

Se consideră o structură "feedforward" perceptron multistrat. Datorită dimensiunii patternurilor de intrare şi de ieşire dorită, rezultă că, dimensiunea straturilor neurale de intrare şi de ieşire este $M = W$, respectiv $N = 1$.

Scanned with OKEN Scanner

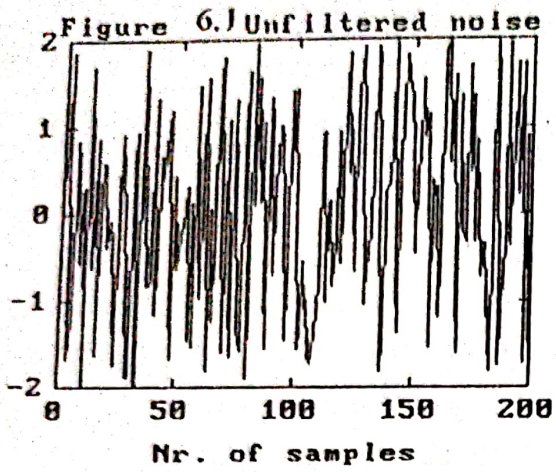


Fig. 6.1 Zgomotul nefiltrat (200 eşant.)

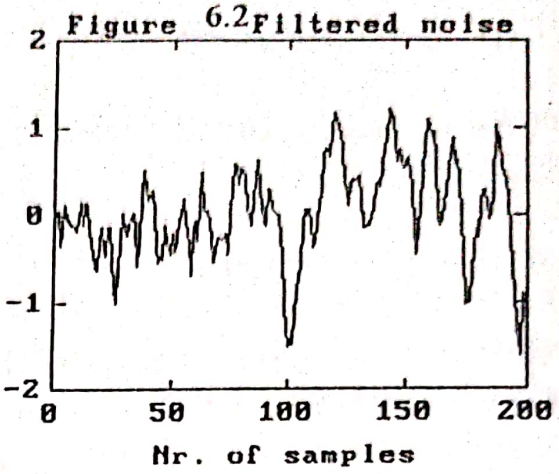


Fig. 6.2 Zgomotul filtrat (200 eşant.)

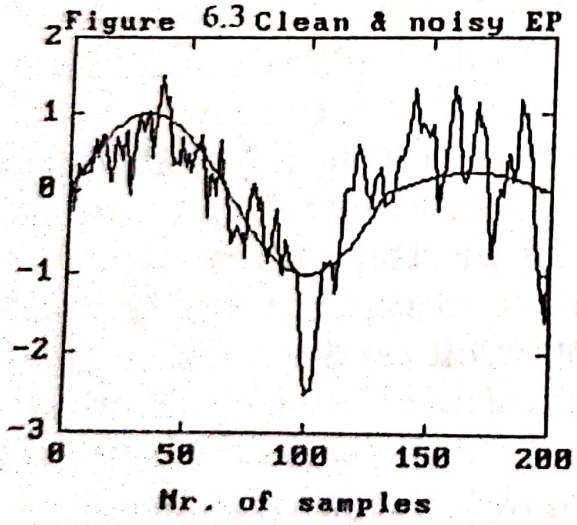


Fig. 6.3 Potenţialul evocat corect şi zgomotos (200 eşantioane)

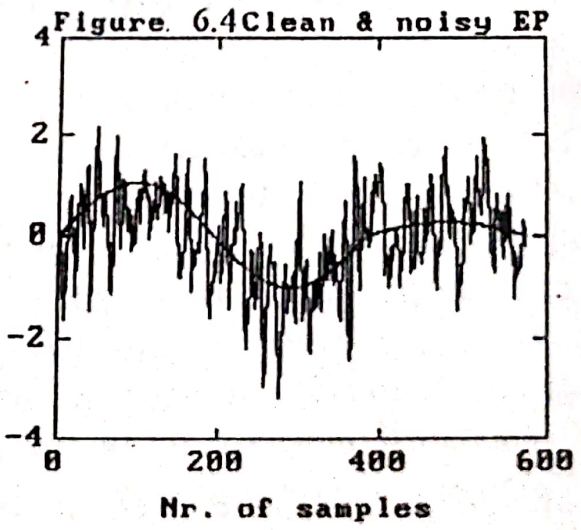


Fig. 6.4 Potenţialul evocat corect şi zgomotos (576 eşantioane)

Scanned with OKEN Scanner

Relaţia Baum – Haussler – Widrow (4.8) pentru calculul numărului de noduri ascunse, pentru $P = 576$, $\varepsilon = 0.15$, $M = 11$ şi $N = 1$, conduce la inegalitatea: $H \leq 7.2$. Dacă se alege dimensiunea stratului ascuns $H = 7$, atunci structura neurală este 11 – 7 – 1 (11 noduri de intrare, 7 noduri ascunse şi 1 nod de ieşire).

❒ ***Algoritmul de învăţare***

Algoritmul de învăţare considerat este BKP [R3, ş.a.] cu moment, prezentat în par. 5.6.1.2, 6.3.1.2 în varianta standard, respectiv în par. 6.3.1.3, 6.3.1.4, 6.3.2.2, şi 6.3.2.3, pentru funcţii cost estimatori robuşti. La intrarea reţelei se prezintă datele din setul de învăţare construit la punctul 6.4.1.1. Funcţia de activare *f(net)* a fiecărui perceptron simplu din stratul ascuns şi de ieşire este tangenta sigmoidală din relaţia (3.10), multiplicată cu o valoare constantă *a*.

6.4.2 Evaluarea performanţelor

Pentru evaluarea performanţelor structurii neurale utilizate pentru estimarea potenţialelor evocate, dintre indicatorii menţionaţi în tab. 5.4, se utilizează: eroarea pătratică medie normată la numărul de patternuri, raportul semnal – zgomot la vârf exprimat în decibeli, numărul operaţiilor efectuate în etapa de test, timpul necesar pentru etapa de test şi spaţiul necesar implementării.

6.4.3 Performanţele reţelelor neurale "feedforward" cu funcţii cost estimatori robuşti de tip M, în varianta de optimizare mono – criteriu

6.4.3.1 Cazul pantei constante a funcţiei de activare

Se consideră rezolvarea problemei de estimare prezentată în par. 6.4.1 pentru cazul funcţiei de activare cu pantă constantă (s = const.) şi funcţii cost estimatori robuşti din tab. 6.1

Experimentele realizate sunt enumerate în tab. 6.3. Primele patru dintre acestea, sunt destinate verificării alegerii corespunzătoare a dimensiunilor structurii. Sunt modificate atât numărul (M) perceptronilor simpli din stratul de intrare (lăţimea ferestrei W), cât şi cel din stratul ascuns H, precum şi numărul patternurilor P prezentate reţelei spre învăţare. Experimentul 5 studiază influenţa alegerii criteriilor de cost diferite norma L_2, asupra învăţării şi performanţelor reţelei.

Scanned with OKEN Scanner

Tab. 6.3 Experimente desfăşurate
$\alpha = 0.1$, $s = 0.5$, $N = 1$, $a = 0.5$, $f(net)$ din relaţia (3.10),
ieşirea dorită = PE corect, semnalul de test = PE cu zgomot

Exp nr.	M (=W)	H	P	Nr. epoci	Prag	Funcţie cost	Semnal PE învăţare
1	11	7	200	variab.	const.	L_2	corect
	11	7	576	variab.	const.	L_2	corect
2	11	7	576	variab.	const.	L_2	corect
	11	7	576	variab.	adaptiv	L_2	corect
3	5,7,9,11,13	7	576	variab.	adaptiv	L_2	corect
4	11	6,7,8	576	variab.	adaptiv	L_2	corect
5	11	7	576	1500	adaptiv	L_2, Huber, Hampel, Logistic	cu zgomot, diferit de cel de test

Tab. 6.4 Influenţa numărului de patternuri P asupra erorii de învăţare

Nr. patternuri	Epoci	Nr. total eşantioane	Eroarea în etapa de învăţare
200	512	102 400	0.0110
576	153	88 128	0.0011

■ Experimentul 1

Se consideră reţeaua cu structura 11 – 7 – 1, care a învăţat (tab. 6.3) patternurile din setul datelor de învăţare. Influenţa numărului de patternuri P asupra erorii este prezentată în tab. 6.4. Se observă că, pentru o mărire de 2.88 ori, a numărului de patternuri, eroarea de învăţare scade de 3.34 ori.

■ Experimentul 2

Curbele de învăţare pentru experimentul 2 (tab. 6.3) sunt prezentate în fig. 6.5. Rezultă că, pragurile şi pantele adaptive măresc viteza de convergenţă a algoritmului.

Scanned with OKEN Scanner

■ Experimentul 3

Reţeaua a învăţat cu un număr diferit de noduri în stratul de intrare (tab. 6.5). S-a impus obţinerea unei valori a erorii de învăţare mai mică de 0.00066. Intrucât valoarea M este chiar dimensiunea ferestrei W, tab. 5.8 prezintă influenţa acesteia asupra vitezei de convergenţă şi a erorii de test.

Dacă valoarea W este prea mică, reţeaua nu poate învăţa relaţiile neliniare dintre eşantionul de date curent şi eşantioanele din afara ferestrei [X2]. Pentru $W = 5$ şi $W = 7$, eroarea de test este mai mică deoarece reţeaua nu realizează o codare internă ($M \leq H$) dar, extragerea semnalului din zgomot nu este satisfăcătoare. Dacă dimensiunea ferestrei W este prea mare ($M = W = 13$), reţeaua încearcă să reţină prea multe corelaţii şi eroarea are o valoare ridicată.

Variaţia raportului semnal – zgomot la vârf (PSNR) şi a numărului epocilor de învăţare, sunt reprezentate grafic în fig. 6.6 pentru valori ale ferestrei W de 5, 7, 9, 11 şi 13.

Tab. 6.5 Influenţa dimensiunii ferestrei W asupra performanţelor reţelei

M (= W)	H	N	Epoci	Nr. total patternuri	Eroare de test
13	7	1	44	25 344	0.1199
11	7	1	69	39 744	0.1118
9	7	1	87	50 112	0.0802
7	7	1	10	5760	0.0597
5	7	1	10	5760	0.0378

■ Experimentul 4

Influenţa numărului de noduri din stratul ascuns asupra erorii în faza de învăţare (experimentul 4, tab. 6.5) este prezentată în fig. 6.7. Deşi adăugarea unui nod ascuns suplimentar creşte viteza de convergenţă, nu îmbunătăţeşte eroarea de test.

In urma experimentelor 1 – 4, se observă că, dimensiunile structurii neurale alese (11 – 7 – 1) reprezintă un compromis din punct de vedere al vitezei de convergenţă şi performanţelor în faza de test, în cazurile studiate. Invăţarea reţelei cu semnale PE corecte (atât semnalul de intrare, cât şi cel de ieşire dorit), şi apoi testarea acesteia pe semnale PE cu zgomot, a dovedit că reţeaua nu obţine performanţele dorite. In fig. 6.8 este prezentat semnalul obţinut la ieşire în faza de test, în comparaţie cu semnalul de ieşire dorit.

Scanned with OKEN Scanner

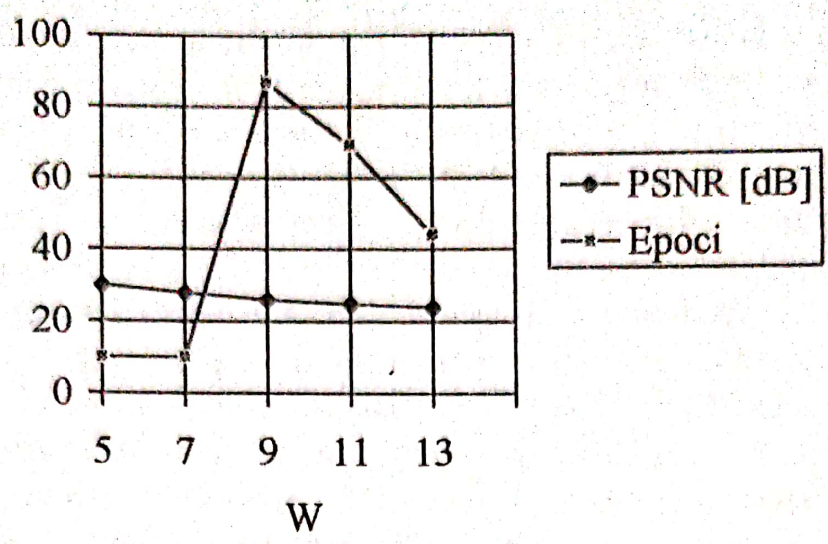


Fig. 6.6 PSNR şi numărul epocilor de învăţare, în funcţie de dimensiunea ferestrei W

■ Experimentul 5

Experimentul 5 (tab. 6.5) a urmărit evaluarea erorii de test pentru cazul alegerii funcţiilor cost estimatori robuşti. Etapa de învăţare a durat 1050 de epoci (s-au prezentat reţelei 604 800 patternuri de intrare). Ponderile iniţiale au avut valori aleatoare, dar aceleaşi pentru toate etapele experimentului. In fig. 6.10 – 6.13 sunt prezentate semnalele de ieşire în faza de test, pentru funcţiile cost din tab. 6.1. Pentru aceleaşi condiţii de învăţare, s-au obţinut performanţe mai bune (valori mai reduse ale erorii medii pătratice de test, tab. 6.6 şi valori mai ridicate ale PSNR, fig. 6.14) pentru criterii de cost diferite de norma L_2 a erorii (tab. 6.6).

De asemenea, experimentul 5 a pus în evidenţă importanţa alegerii valorii parametrului β (din expresiile funcţiilor cost). Pentru o valoare mare a acestuia, algoritmul de învăţare este echivalent algoritmului BKP standard, cu normă L_2 a erorii. Dacă β are o valoare mică, atunci prima derivată a funcţiei eroare aproximează bine funcţia semn. Dacă β are o valoare foarte mică, există cazuri în care reţeaua nu învaţă, (de exemplu, dacă $\beta \leq 0.1$, atunci reţeaua nu poate depăşi prima epocă, deoarece $e_j(\xi) > \beta$ şi ponderile conexiunilor dintre stratul ascuns şi cel de ieşire nu se modifică). Semnalul obţinut la ieşire este prezentat în fig. 6.13.

O îmbunătăţire semnificativă a performanţei se obţine în cazul medierii semnalului de ieşire, pe un ansamblu compus din câteva reţele (exemplu: fig. 6.12, pentru funcţia logistică, 5 reţele).

Scanned with OKEN Scanner

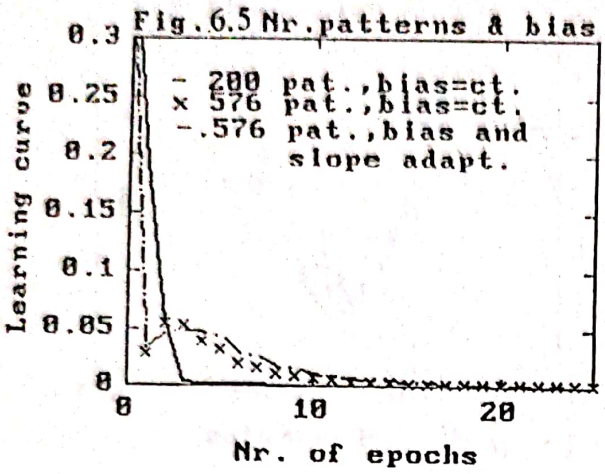


Fig. 6.5 Influenţa numărului de patternuri

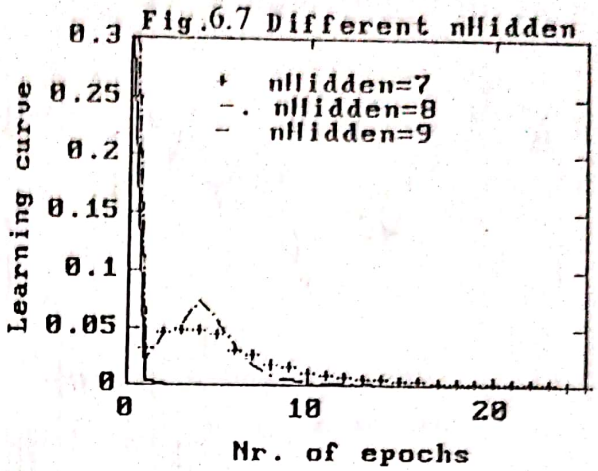


Fig. 6.7 Influenţa numărului diferit de noduri ascunse

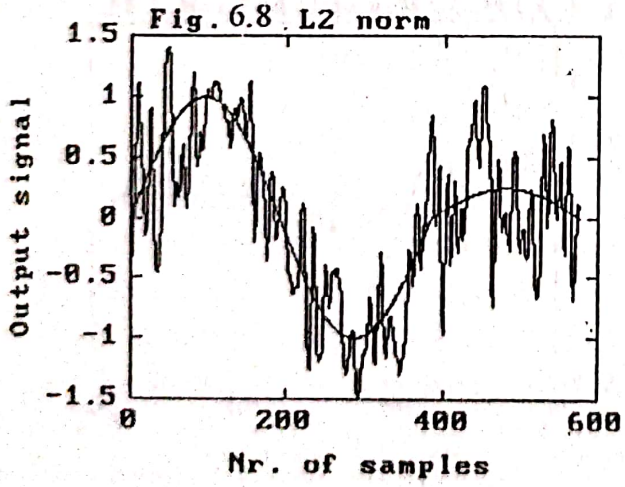


Fig. 6.8 Potenţialul evocat obţinut în cazul normei L_2

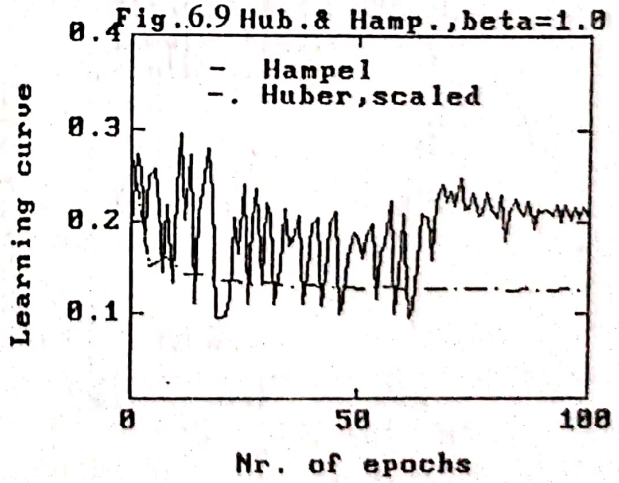


Fig. 6.9 Curba de învăţare pentru funcţiile Huber, Hampel

Scanned with OKEN Scanner

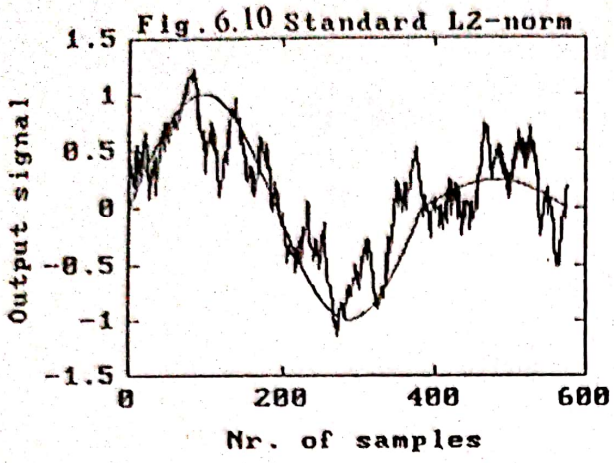


Fig. 6.10 Potenţialul evocat estimat (norma L_2, exp.5)

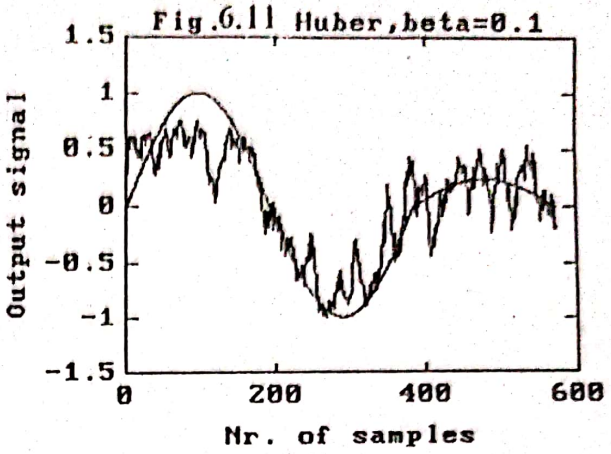


Fig. 6.11 Potenţialul evocat estimat (funcţia Huber, exp. 5)

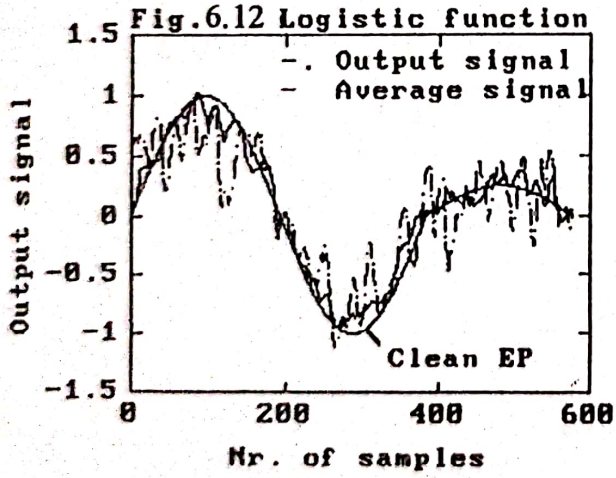


Fig. 6.12 Potenţialul evocat estimat (funcţia logistică, exp. 5)

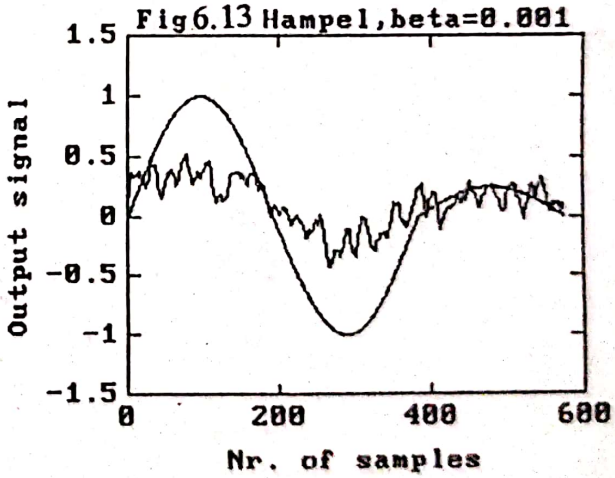


Fig. 6.13 Potenţialul evocat estimat (funcţia Hampel, exp. 5)

Scanned with OKEN Scanner

Tab. 6.6 Eroarea pătratică medie normată la ieşire, pentru diferite funcţii cost

Funcţia cost	Eroarea pătratică medie de test
Norma L_2	0.09040
Huber, $\beta = 0.01$	0.08912
Logistic, $\beta = 0.001$	0.08705

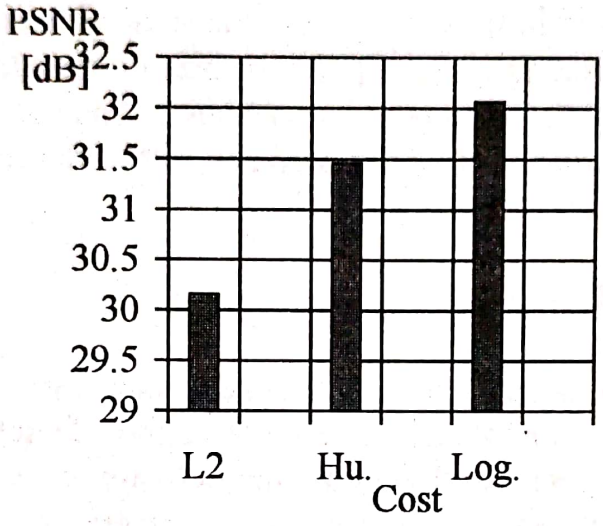


Fig. 6.14 PSNR la ieşirea reţelei în etapa de test, pentru funcţiile cost L_2, Huber (Hu., cu $\beta = 0.01$) şi logistică (Log., cu $\beta = 0.001$)

Repetarea experimentului 5 (tab. 6.5), alegând ca valoare a ieşirii dorite la fiecare pas, pe cea imediat următoare ferestrei de intrare (exemplu: dacă valorile de intrare sunt valorile PE cu zgomot x_1, ..., x_{11}, atunci ieşirea dorită este valoarea corectă x_{12}) arată că, performanţele reţelei nu se îmbunătăţesc datorită acestei învăţări "cu context". Singurele avantaje sunt, pe de o parte posibilitatea de alegere a dimensiunii ferestrei ($W = M$) ca valoare pară sau impară, precum şi funcţionarea reţelei ca predictor, fiind capabilă în acest caz, să obţină la ieşire valoarea următoare, pe baza a 11 valori anterioare.

6.4.3.2 Cazul pantei variabile a funcţiei de activare

Se consideră rezolvarea problemei de estimare pentru cazul funcţiei de activare cu pantă variabilă (s = adaptivă) şi funcţii cost estimatori robuşti (tab. 6.1).

Scanned with OKEN Scanner

Fie structura neurală aleasă în par. 6.4.1.2. Funcţia de activare a perceptronului simplu este $f_1(s, net)$ dată de relaţia (3.9) multiplicată cu constanta $a = 0.5$, sau $f_2(s, net)$ (funcţia Fahlman) din relaţia (3.11), ambele cu panta adaptivă s. Funcţiile cost estimatori robuşti sunt alese din tab. 6.1.

Semnalul PE corect şi cel zgomotos pentru $P = 576$ eşantioane, sunt prezentate în fig. 6.15. Experimentele realizate sunt menţionate în tab. 6.7.

Reţelele au învăţat 400 de epoci (un număr total de 230 400 de patternuri de intrare). Ponderile şi pantele iniţiale au fost alese aleator, dar aceleaşi pentru toate etapele experimentului. Pantele iniţiale ale funcţiilor de activare aparţin intervalului [0.1, 1.0], iar ponderile iniţiale aparţin intervalului [–0.3, +0.3].

Valorile parametrilor de învăţare, valorile minimă şi maximă impuse pantei, precum şi eroarea pătratică minimă obţinută în faza de test, în sensul normei L_2, sunt prezentate în tab. 6.7. Variaţia valorii PSNR în funcţie de criteriul de cost ales, pentru fiecare dintre funcţiile de activare este reprezentată grafic în fig. 6.16.

■ Experimentul 6

In experimentul 6, s-au obţinut performanţe mai bune pentru funcţia cost normă L_2 a erorii, folosind pante adaptive ale funcţiilor de activare, faţă de cazul pantelor constante. Atunci când funcţia de activare este $f_2(s, net)$, eroarea descreşte mai lent decât pentru $f_1(s, net)$ (conform relaţiei (3.11)). Curbele de învăţare pentru experimentul 6 sunt prezentate în anexa 5 (fig. A5.1, A5.2).

Tab. 6.7 Influenţa pantelor adaptive asupra estimării potenţialelor evocate
400 epoci, $\eta = 0.01$, $\alpha = 0.1$, $\rho = 0.0001$, $s_{min} = 0.3$, $s_{max} = 1.5$,
semnalul de ieşire dorit = semnal PE corect,
semnal de test = semnal PE cu zgomot

Exp nr.	Funcţia cost	Funcţia de activare	β	γ	Eroare de test (L_2)
1	Norma L_2	$f_1(s, net)$, $s = 0.5$ = const.	—	—	0.067680
		$f_1(s, net)$, s = adaptiv	—	0.00001	0.064606
		$f_2(s, net)$, s = adaptiv	—	0.00001	0.065040
2	Hampel	$f_1(s, net)$, s = adaptiv	0.50	0.00001	0.071317
		$f_2(s, net)$, s = adaptiv	0.50	0.00001	0.070140
3	Logistică	$f_1(s, net)$, s = adaptiv	0.10	0.00001	0.067472
			0.01	0.00001	0.070420
		$f_2(s, net)$, s = adaptiv	0.10	0.00001	0.066940
			0.01	0.00001	0.070620

Scanned with OKEN Scanner

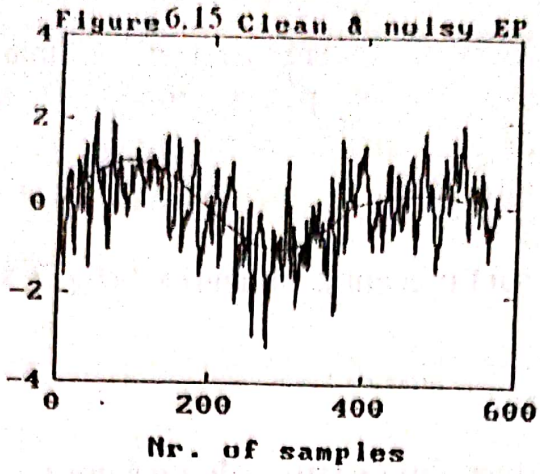


Figura 6.15 Potenţialul evocat corect şi potenţialul evocat cu zgomot

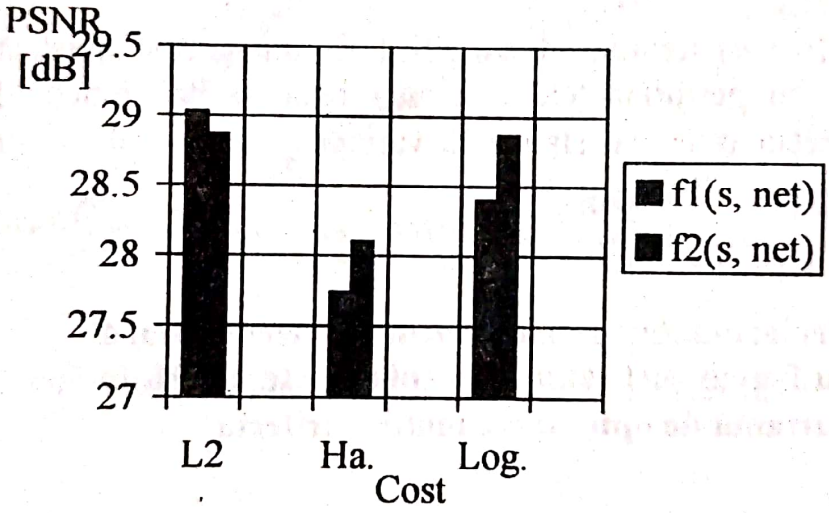


Fig. 6.16 PSNR la ieşirea reţelei în etapa de test, pentru funcţiile cost L_2, Hampel (Ha., cu $\beta = 0.5$) şi logistică (Log., cu $\beta = 0.1$), când s = adaptiv

■ **Experimentul 7**

In cazul funcţiei cost Hampel, eroarea de test după 400 de epoci de învăţare este mai mare decât pentru norma L_2 a erorii. Pentru aceeaşi valoare a parametrului $\beta = 0.5$, funcţia de activare $f_2(s, net)$ conduce la rezultate mai bune (tab. 6.7).

Curbele de învăţare sunt prezentate în anexa 5 (fig. A5.3, A5.4).

Scanned with OKEN Scanner

■ **Experimentul 8**

Pentru funcţia cost logistică, eroarea de test este mai mare decât în cazul normei L_2, dar mai scăzută decât pentru funcţia Hampel. Pentru aceeaşi valoare $\beta = 0.1$, funcţia de activare $f_2(s, net)$ conduce la valori mai scăzute ale erorii decât $f_1(s, net)$. Pentru $\beta = 0.01$, rezultatul pentru $f_2(s, net)$ este mai slab decât pentru $f_1(s, net)$ (tab. 6.7).

Curbele de învăţare sunt prezentate în anexa 5 (fig. A5.5, A5.6).

■ **Experimentul 9**

Studiul influenţei termenului negativ γ de învăţare a pantei adaptive, pentru funcţiile de activare a nodului $f_1(s, net)$ şi $f_2(s, net)$ a condus la propunerea unei noi strategii de învăţare, numită "flip – flop" [D204, L205]. Rezultatele numerice şi grafice, precum şi interpretarea acestora, sunt prezentate în anexa 5.

Pentru funcţia de activare $f_1(s, net)$, ierarhia performanţelor este: ① norma L_2, ② funcţia cost logistică, ③ funcţia Hampel, cu rezultatele cele mai bune pentru criteriul L_2.

Pentru $f_2(s, net)$, ierarhia se modifică: ① funcţia cost logistică, ② Hampel, ③ norma L_2, cu performanţele cele mai bune pentru funcţia logistică. De asemenea, aceasta este mai stabilă la variaţiile parametrului γ (anexa 5: tab. A5.1, A5.2).

6.4.4 Performanţele reţelelor neurale "feedforward" cu funcţii cost estimatori robuşti de tip M, în varianta de optimizare multi – criteriu

6.4.4.1 Cazul pantei constante a funcţiei de activare

Se consideră reţeaua neurală "feedforward" cu structura 11 – 7 – 1, funcţia de activare a nodului $f_1(net)$ şi funcţia cost multi – criteriu dată de relaţia (6.63), unde funcţiile estimatori robuşti sunt norma L_2 şi funcţia logistică. La intrarea reţelei se prezintă în etapa de învăţare semnalul potenţial evocat cu zgomot, iar în etapa de test semnalul potenţial evocat cu zgomot, diferit de semnalul de învăţare. Parametrii de învăţare sunt aceeaşi ca în tab. 6.3.

Dependenţa parametrului $1 - \theta$ de valorile $c > 0$ este reprezentată grafic în fig. 6.17. Se observă că, pentru $c = 0.8$, se obţine variaţia dorită crescătoare, de la valoarea 0 la 1 (respectiv, pentru θ, descrescătoare de la valoarea 1 la 0).

Scanned with OKEN Scanner

Pentru cazul în care parametrul θ se modifică pe baza relaţiei (6.66), calculul mediei semnalului de test conduce la valoarea $\mu_{x\ test} = 0.76 \cong 0.8$, confirmând strategia propusă. Eroarea de test ca normă L_2 a erorii este EPMN = 0.0585 şi PSNR = 39.240 dB, indicând performanţe superioare celor obţinute în paragrafele anterioare. Reprezentarea semnalului de ieşire obţinut în etapa de test, a semnalului de ieşire dorit şi a erorii pătratice medii la fiecare epocă (fig. 6.18) confirmă acest lucru.

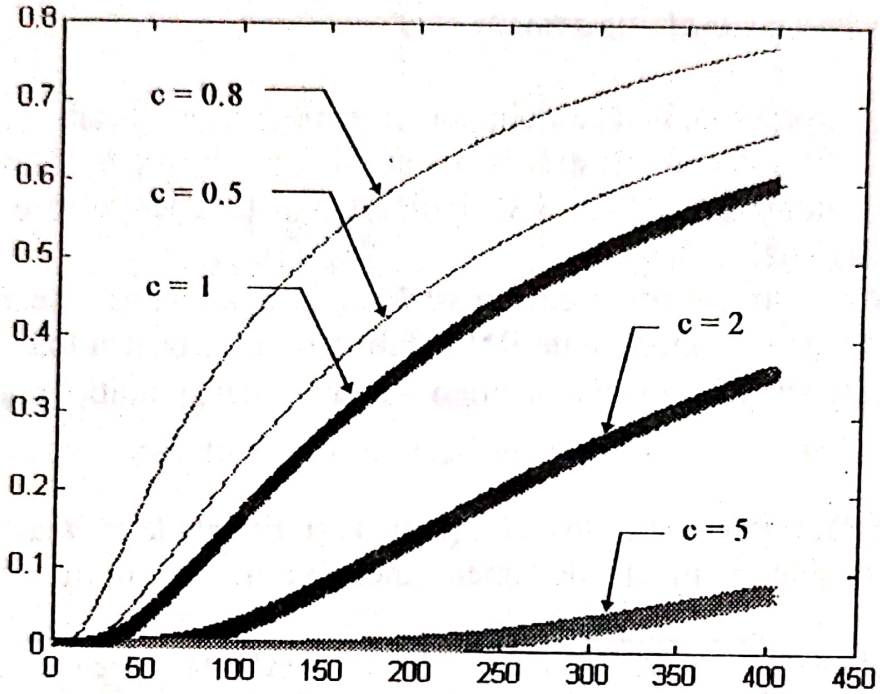


Fig. 6.17 Modificarea parametrului $1 - \lambda$, pentru diferite valori $c > 0$

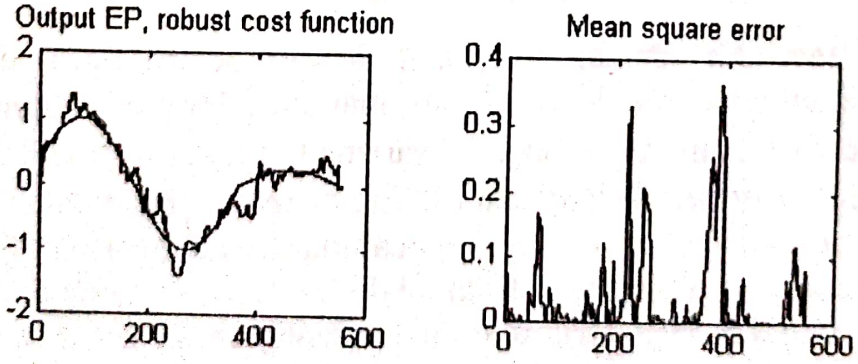


Fig. 6.18 Semnalul de ieşire obţinut în etapa de test, semnalul de ieşire dorit şi eroarea pătratică medie, pentru cazul funcţiei cost estimator robust, în varianta de optimizare multi – criteriu

Scanned with OKEN Scanner

6.4.5 Interpretarea comparativă a rezultatelor experimentale obţinute cu funcţii cost estimatori robuşti de tip M

Pe baza indicatorilor menţionaţi în par. 6.4.2, se poate realiza comparaţia rezultatelor experimentale obţinute în variantele mono – criteriu şi multi – criteriu.

❒ ***Eroarea de învăţare, eroarea de test şi raportul semnal – zgomot la vârf***

Compararea erorilor de învăţare obţinute în experimentul 3 (tab. 6.5) pentru diferite valori ale dimensiunii ferestrei W, cu erorile obţinute de un filtru median având de aceeaşi dimensiune (tab. 6.8), indică faptul că RNA obţine rezultate mai bune în fiecare caz.

EPMN pentru filtrul median cu fereastră de 11 eşantioane este mai mare decât toate erorile de test obţinute de RNA (tab. 6.6, 6.7), pentru toate funcţiile cost estimatori robuşti, atât în varianta mono – criteriu, cât şi multi – criteriu.

Tab. 6.8 EPMN în etapa de învăţare (exp. 3) şi EPMN la ieşirea filtrului median, în funcţie de dimensiunea ferestrei de intrare

Dimensiunea (W) ferestrei de intrare	EPMN în etapa de învăţare RNA	EPMN la ieşirea filtrului median
11	0.1118	0.16667
9	0.0802	0.1905
7	0.0597	0.2208
5	0.0378	0.2612

Valoarea raportului semnal – zgomot la vârf pentru filtrul median cu fereastră de 11 eşantioane este de 31.778 dB, mai mică decât cea obţinută pentru RNA cu funcţie cost estimator robust, în variantele mono – criteriu şi multi – criteriu. Obţinerea unor performanţe ale filtrului median comparabile cu cele ale RNA cu funcţii cost estimatori robuşti, implică utilizarea dimensiunii ferestrei de intrare mai mare de 17 eşantioane, dar timpul de test creşte considerabil.

Semnalul obţinut la ieşirea filtrului median cu fereastră de 3, 5, 7, 9 şi 11 eşantioane, semnalul de ieşire dorit şi eroarea pătratică medie pentru fiecare eşantion, sunt reprezentate în fig. 6.19 – 6.21. Din comparaţia cu semnalele obţinute în etapa de test de RNA (fig. 6.10 – 6.13, 6.18) se observă că, rezultatele reţelei neurale utilizate sunt mai bune decât cele ale filtrului median.

Scanned with OKEN Scanner

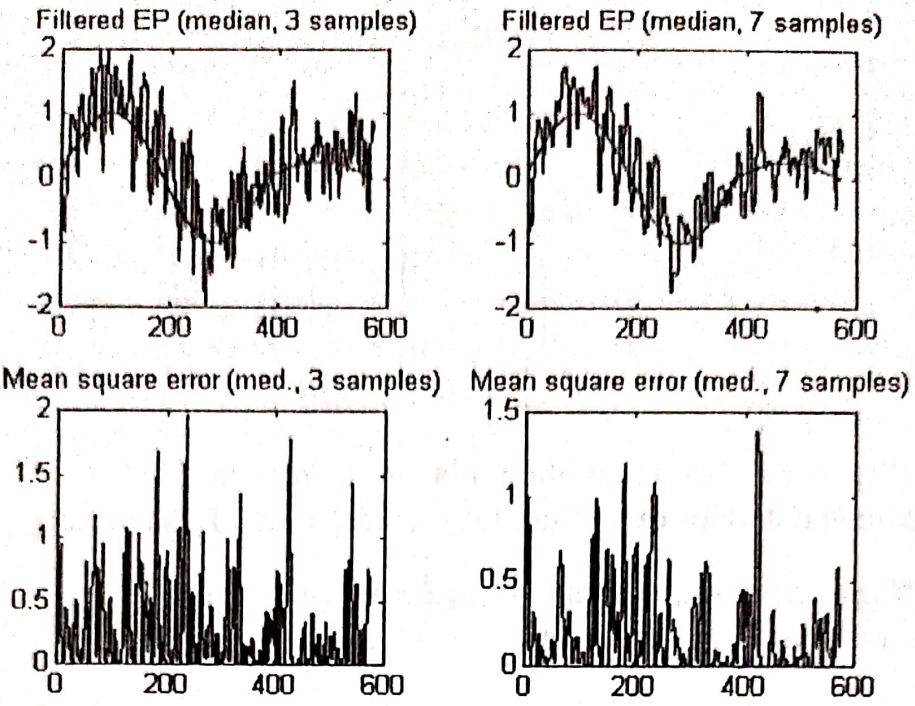


Figura 6.19 Semnalul de ieşire obţinut, semnalul de ieşire dorit şi eroarea pătratică medie, pentru filtrul median cu fereastra de 3 şi 7 eşantioane

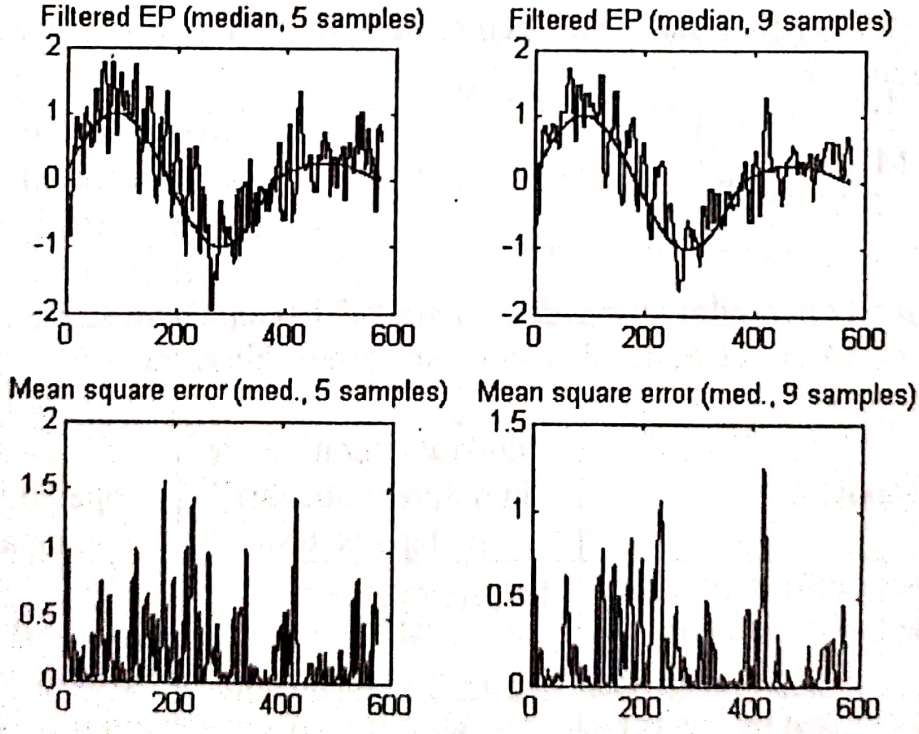


Figura 6.20 Semnalul de ieşire obţinut, semnalul de ieşire dorit şi eroarea pătratică medie, pentru filtrul median cu fereastra de 5 şi 9 eşantioane

Scanned with OKEN Scanner

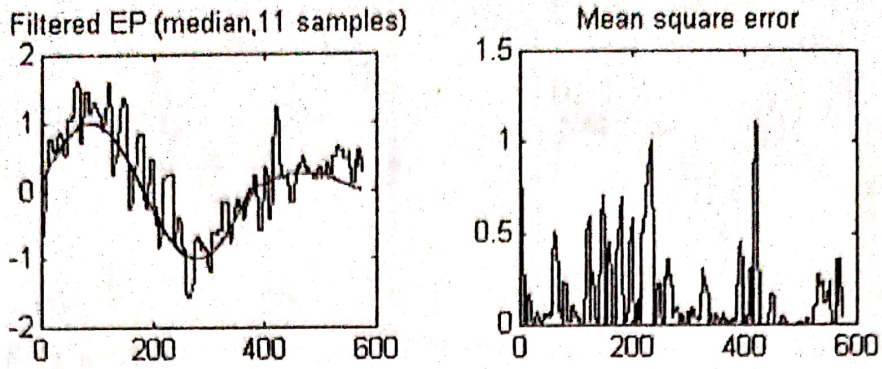


Fig. 6.21 Semnalul obţinut la ieşire, în cazul filtrării mediene a datelor de intrare, prin fereastră de 11 eşantioane

❒ ***Numărul operaţiilor de "înmulţire + adunare" efectuate în etapa de test***

Numărul operaţiilor de "înmulţire + adunare" efectuate în etapa de test, calculat cu relaţia (4.223), este aproximativ acelaşi pentru cazul utilizării funcţiilor cost în varianta mono – criteriu, dar creşte de mai mult de două ori în varianta multi – criteriu (tab. 6.9).

Filtrul median având aceeaşi dimensiune a ferestrei de intrare ca şi RNA realizează pentru cele P patternuri de intrare un număr de operaţii de comparare dat de relaţia (6.70) [P17], iar pentru pentru $M = W = 11$ şi $P = 576$ se obţine rezultatul din tab. 6.9.

$$op_{comp} = \frac{M(M-1)}{2} P \qquad (6.70)$$

Tab. 6.9 Numărul operaţiilor "înmulţire + adunare" în etapa de test, pentru RNA şi numărul operaţiilor de comparare pentru filtrul median

Modelul	Numărul operaţiilor de "înmulţire + adunare" în etapa de test	Numărul operaţiilor de comparare
RNA cu funcţii cost estimatori robuşti de tip M, varianta mono – criteriu	48384	0
RNA cu funcţii cost estimatori robuşti de tip M, varianta multi – criteriu	96769	0
Filtrul median	0	31680

Scanned with OKEN Scanner

❒ *Timpul necesar pentru etapa de test*

Duratele etapelor de testare, atât calculate cât şi măsurate, pentru RNA cu funcţii cost estimatori robuşti în variantele mono – criteriu şi multi – criteriu, precum şi pentru filtrul median cu dimensiunea ferestrei $W = 11$, sunt prezentate în tab. 6.10. Evaluarea s-a realizat pe un calculator IBM – PC 486 / 66 MHz. S-a notat cu t_0 durata unei operaţii simple de adunare sau înmulţire, cu t_t timpul de transfer al unei date reprezentate pe $b = 16$ biţi. S-a presupus un raport υ unitar între numărul biţilor folosiţi pentru reprezentare şi valoarea b.

Durata timpului de test pentru filtrul median este [P17]:

$$t_{test\ med} = M\ P\ t_{comp} \tag{6.71}$$

unde t_{comp} este timpul necesar unei singure comparaţii pentru ordonarea datelor.

Se observă că, RNA funcţionează mai rapid în etapa de test decât filtrul median. Mărirea dimensiunilor ferestrei de intrare a acestuia, pentru a obţine valori PSNR comparabile cu cele ale reţelei neurale, creşte timpul de test (pentru $W = 17$ eşantioane, timpul de test este 22 s).

Tab. 6.10 Duratele etapelor de testare pentru RNA şi filtrul median

Modelul	Durata calculată a etapei de test	Durata măsurată a etapei de test [s]
RNA cu funcţii cost estimatori robuşti, în varianta mono – criteriu	$1152\ (11\ t_t + 18\ t_0)$	4
RNA cu funcţii cost estimatori robuşti, în varianta multi – criteriu	$1152\ (11\ t_t + 25\ t_0)$	9
Filtru median	$6336\ t_{comp}$	10

❒ *Spaţiul necesar implementării*

Spaţiul necesar implementării structurii neurale nu depinde de funcţia cost aleasă. Pentru un nod ascuns este valabilă relaţia (5.49), multiplicată cu numărul nodurilor ascunse ale structurii.

Scanned with OKEN Scanner

6.5 CONCLUZII, DIRECŢII DE STUDIU ŞI PERSPECTIVE

Performanţele RNA pentru funcţii cost estimatori robuşti, în variantele mono– şi multi – criteriu, au constituit obiectul studiului prezentului capitol. Alegerea acestora este motivată pe de o parte de utilizarea seturilor de date cu zgomot, cu distribuţie necunoscută sau amestec de distribuţii nestandard şi, pe de altă parte, de existenţa datelor "outliers" în seturile menţionate.

Problema abordată a fost estimarea potenţialelor evocate, pentru cazul unui semnal generat artificial.

6.5.1 Concluzii teoretice

Proiectarea RNA folosind funcţii cost estimatori robuşti de tip M în varianta de optimizare mono – criteriu impune rescrierea relaţiilor matematice de modificare a ponderilor în algoritmul BKP. Acest pas a fost realizat atât în cazul pantei constante, cât şi al pantei variabile a funcţiei de activare a nodului.

In cazul proiectării cu funcţii cost estimatori robuşti de tip M în varianta de optimizare multi – criteriu, rezolvarea teoretică se bazează pe utilizarea metodei normei L_p sau a metodei cu constrângeri ε.

6.5.2 Concluzii experimentale

Studiul experimental realizat a fost constituit din următoarele etape:

❶ Formularea problemei de studiu (estimarea potenţialelor evocate).

❷ Construirea setului datelor de învăţare şi de test. Semnalul potenţial evocat corect are forma unei sinusoide complete, urmată de o jumătate de sinusoidă atenuată. Semnalul potenţial evocat cu zgomot este constituit din semnalul corect, peste care s-a suprapus zgomot de impulsuri.

❸ Alegerea structurii de reţea "feedforward" perceptron multistrat, cu dimensiunile 11 – 7 – 1, aplicând relaţia Baum – Haussler – Widrow (4.8).

❹ Intrucât după învăţarea semnalelor PE corecte, structura neurală a obţinut performanţe nesatisfăcătoare pentru semnale de test cu zgomot de impulsuri, a fost necesară reluarea învăţării pentru semnale de intrare PE cu zgomot. S-a efectuat testarea pe semnale PE cu zgomot, diferite de cele din etapa de învăţare. Funcţia cost aleasă a fost norma L_2 a erorii, iar panta funcţiei de activare a perceptronului simplu a fost constantă.

❺ Studiul efectului dimensiunii ferestrei (numărului de perceptroni simpli

Scanned with OKEN Scanner

din stratul de intrare), al numărului celor din stratul ascuns şi al numărului de patternuri din setul de date de învăţare, asupra vitezei de convergenţă şi erorii de test. Funcţia cost a fost norma L_2 a erorii, iar panta funcţiei de activare a perceptronului simplu a fost constantă.

❻ Pentru aceiaşi parametri de învăţare şi număr de patternuri, studiul efectului alegerii funcţiilor cost estimatori robuşti de tip M, în varianta mono – criteriu, asupra performanţelor RNA [D201, L205].

In cazul pantei constante a funcţiei de activare a nodului, ierarhia performanţelor este: ① funcţia cost logistică, ② funcţiile cost Huber şi L_2, ③ funcţia Hampel. Imbunătăţirea performanţelor se realizează în continuare, prin medierea rezultatelor pe ansambluri compuse din numai câteva reţele. Valorile parametrului β influenţează rezultatele obţinute.

In cazul pantei adaptive a funcţiei de activare a nodului [D204, L205], pentru structura neurală care a învăţat semnale PE cu zgomot şi a fost testată pe alte semnale PE cu zgomot, rezultatele experimentale mai bune obţinute în cazul alegerii funcţiilor cost estimatori robuşti de tip M şi a pantelor constante ale funcţiilor de activare, nu se pot extinde. Pentru a reconstrui o ierarhie a performanţelor, este necesară alegerea perechilor {funcţie cost, funcţie de activare}, întrucât combinaţii diferite ale acestora conduc la valori diferite ale erorilor de test.

❼ Studiul efectului alegerii funcţiilor cost estimatori robuşti de tip M, în varianta multi – criteriu, asupra performanţelor RNA, în cazul pantei constante a funcţiei de activare a nodului. Rezultatele obţinute sunt mai bune decât cele obţinute în varianta mono – criteriu.

❽ Compararea cantitativă a rezultatelor experimentale obţinute în cazul utilizării funcţiilor cost estimatori robuşti de tip M, în variantele mono – şi multi – criteriu.

Erorile de învăţare şi de test obţinute în ambele variante, sunt mai reduse decât erorile la ieşirea unui filtru median, pentru toate dimensiunile ferestrei de intrare considerate.

PSNR obţinut de RNA în varianta multi – criteriu este mai mare decât cel obţinut în varianta mono – criteriu, ambele fiind superioare valorii PSNR la ieşirea filtrului median.

Numărul operaţiilor de "înmulţire + adunare" în etapa de test se dublează în cazul utilizării funcţiilor cost estimatori robuşti în varianta multi – criteriu, faţă de varianta mono – criteriu. Filtrul median necesită efectuarea operaţiilor de comparare.

Evaluarea timpului necesar pentru etapa de test în cazurile mono– şi multi – criteriu, precum şi pentru filtrul median, pe un calculator IBM – PC 486 / 66 MHz indică faptul că, timpul de test minim este consumat de varianta mono –

Scanned with OKEN Scanner

criteriu, urmată de cea multi – criteriu, diferenţa dintre acestea nefiind însă semnificativă. Filtrul median necesită intervalul de timp maxim în etapa de test.

6.5.3 Direcţii de studiu şi perspective

Printre aspectele care constituie probleme de studiu în continuare, se numără:

❶ Stabilirea criteriilor teoretice pentru alegerea valorilor parametrilor β şi γ, anterior determinate experimental. De asemenea, este necesară stabilirea valorilor optimale ale parametrilor β, γ, în condiţiile în care, în prealabil, se determină rata de învăţare optimală η.

❷ S-a considerat că, panta funcţiei de activare poate varia între valorile pozitive s_{min} şi s_{max}. Este posibilă însă, alegerea acestora şi în domeniul valorilor negative. Intrucât oricare dintre funcţiile nemonotone ce caracterizează modelele neurale de tip RBF, modelele fuzzy, se pot obţine prin suprapunerea funcţiilor de activare sigmoide, cu pante pozitive şi, respectiv, negative, rezultă că abordarea prezentată realizează conexiunea cu aceste două modelele, făcând posibilă dezvoltarea subiectului de studiu.

❸ Alegerea criteriului de cost rămâne o problemă deschisă, fiind dependentă de aplicaţia considerată. Utilizarea estimatorilor robuşti de tip M, în variantele mono – sau multi – criteriu deschide perspective teoretice şi experimentale promiţătoare.

6.6 REZUMATUL CAPITOLULUI

In cap. 6, după precizarea modalităţilor de alegere a funcţiilor cost pentru structuri neurale "feedforward", s-au studiat performanţelor acestora pentru funcţii cost estimatori robuşti, în variantele mono– şi multi – criteriu. Verificarea experimentală a rezultatelor teoretice obţinute în cazul estimării potenţialelor evocate a fost urmată de comparaţia realizată cu cele obţinute de filtrul median.

Scanned with OKEN Scanner

7 OBSERVAȚII FINALE, CONCLUZII ŞI PERSPECTIVE

7.1 OBSERVAȚII FINALE ŞI CONCLUZII

Obiectivul lucrării a fost studiul proiectării RNA "feedforward", utilizate ca instrumente de PDS. Direcţiile în care s-au concentrat cercetările teoretice şi experimentale au fost: proiectarea topologică şi proiectarea cu funcţii cost diferite de norma L_2 a erorii.

Capitolele 1 – 4 prezintă sintetic elementele necesare proiectării: caracteristicile RNA ca instrumente de PDS (cap. 1), stadiul actual al dezvoltării structurilor şi aplicaţiilor RNA în PDS (cap. 2), bazele teoretice (cap. 3), metode de proiectare a RNA "feedforward" (cap. 4). Lucrarea conţine, de asemenea, o serie de contribuţii originale referitoare la proiectarea topologică (metoda mixtă de proiectare a filtrelor neurale din par. 4.7.2, metodele "distructive" cu constrângeri de simetrie din cap. 5) şi la proiectarea cu funcţii cost estimatori robuşti de tip M, în variantele de optimizare mono – şi multi – criteriu (cap. 6).

In final, se impune luarea deciziei de acceptare sau rejectare a RNA proiectate ca instrumente de PDS, în comparaţie cu tehnici clasice de prelucrare (tab. 7.1). Deşi, pe baza rezultatelor teoretice şi experimentale obţinute în problemele de studiu abordate, se poate decide acceptarea acestora ca instrumente performante de PDS, rămân încă aspecte nerezolvate, care reprezintă subiecte ale cercetării ulterioare.

Tab. 7.1 Criterii de acceptare a RNA ca instrument de PDS

Intrebare	Răspuns
Utilizarea RNA realizează o diferenţă semnificativă din punct de vedere al performanţelor, în aplicaţiile considerate sau, poate furniza o reducere semnificativă a costului implementării, fără degradarea considerabilă a performanţelor ?	Da
Proiectarea este simplă, comparativ cu alte metode clasice [H56] ? ▪ format optim intrare – ieşire (Nu totdeauna) ▪ timp de învăţare (Minute) ▪ cerinţe de preprocesare (Depind de aplicaţie) ▪ optimalitate din punct de vedere matematic (Nu) ▪ performanţa estimată (Bună) ▪ cerinţe pentru depanarea sistemului (Reduse)	Da
Prin utilizarea RNA, se poate rezolva problema de PDS pentru care nu există alte metode (standard) ?	?

Scanned with OKEN Scanner

7.2 DIRECŢII DE STUDIU ŞI PERSPECTIVE

❶ Procedurile de simetrizare a structurii neurale (cap. 5) se bazează pe aproximaţii care au scopul evitării creşterii costului implementării, din punct de vedere al timpului de calcul şi al spaţiului de stocare al parametrilor. Imbunătăţirea acestor aproximări, în scopul reducerii erorii, reprezintă o problemă deschisă.

❷ Continuarea studiului teoretic al problemei de reprezentare, pentru stabilirea limitelor în care o structură neurală "feedforward" parţial interconectată poate aproxima o relaţie intrare – ieşire dată considerată.

❸ Simetrizarea structurilor neurale prin metodele propuse, urmată de reprezentarea datelor în planul complex [D205] fac posibilă simplificarea etapei de extragere a regulilor necesare unui sistem expert. Intrucât realizarea conectării RNA cu alte module (expert, fuzzy, etc.) şi încadrarea în sisteme de prelucrare de mari dimensiuni este o problemă esenţială pentru aplicaţiile industriale, contribuţiile lucrării reprezintă paşi în această direcţie.

7.3 ÎN LOC DE ÎNCHEIERE

Chiar dacă instrumentele de studiu utilizate în domeniul RNA în general şi în lucrare în particular, sunt preluate din matematică, fizică, ingineria calculatoarelor, etc., rezultatul se caracterizează în primul rând printr-o perspectivă diferită asupra problemelor anterioare, din care apar noi soluţii, precum şi noi probleme. Este evident că, reţelele neurale artificiale şi-au găsit locul potrivit ca instrument de prelucrare a semnalelor. Este adevărat că, "aşa cum nu există în prezent departamente de ecuaţii diferenţiale la universităţi, probabil că în final nu vor mai exista departamente specializate de studiu al reţelelor neurale. Dar, tot aşa cum [...] se studiază ecuaţiile diferenţiale, [vom] continua să studiem RNA, ghidaţi de rezultate şi de înţelegere" (Andreas Weigend).

Pe de altă parte dacă, în finalul lucrării, acceptăm formularea cunoscută a dilemei cercetătorului ("indiferent cât realizezi, niciodată nu este destul; ceea ce nu ai făcut, este totdeauna mai important decât ceea ce ai făcut" [D37]), atunci soluţia continuării studiului este acceptarea provizoratului şi dimensiunii finite a rezultatelor obţinute. "Responsabilitatea schimbării stă în noi", spunea Alvin Toffler, "trebuie să începem cu noi, să învăţăm să nu ne închidem prematur minţile la tot ce este nou, surprinzător, aparent radical". Din această perspectivă, departe de a fi un pas încheiat, consider acest sfârşit al prezentului studiu drept un nou început şi totodată, o încercare în drumul lung, de depăşire a condiţiei de "spirit pedestru".

Scanned with OKEN Scanner

ANEXA 1

SUMAR AL PROGRAMELOR UNIVERSITARE ŞI INDUSTRIALE DE CERCETARE ÎN DOMENIUL REŢELELOR NEURALE ARTIFICIALE (1990 – 1996)

Un sumar al programelor de cercetare în domeniul reţelelor neurale, atât în universităţi, cât şi în companii industriale din întreaga lume, este concentrat în tab. A1.1 – A1.2. Ordonarea este realizată atât după prezenţa în mediul universitar / industrial, cât şi alfabetic. Numele persoanelor menţionate indică faptul că, acestea au avut contribuţii majore la proiectele amintite, chiar dacă ulterior, aceste persoane au continuat sau nu, să funcţioneze în cadrul universităţilor / companiilor respective.

Sinteza a fost realizată pe baza informaţiilor disponibile în perioada 1990 – 1996. Subliniem faptul că, aceasta este departe de a fi exhaustivă, numărul mare al programelor de cercetare în domeniul RNA existente făcând dificilă realizarea unui astfel de obiectiv. Pentru detalii privind multe alte aplicaţii izolate ale RNA, se pot consulta titlurile incluse în bibliografia prezentei lucrări.

Scanned with OKEN Scanner

Tab. A1.1 Proiecte în domeniul rețelelor neurale, în mediul industrial (1990 – 1996)

Compania	Statul	Autori	Problema teoretică abordată
AI Forum, Silicon Valley	SUA		Aspecte teoretice ale inteligenței
Air Force Wright Aeronautical Laboratories	SUA	A. Harry Klopf	Mecanismele biologice ale învățării
AT & T Laboratories	SUA	David W. Tank Sara A. Solla, J. Denker	Modelare cu RNA Generalizare
ATR	Japonia	B. Irie K. Funahashi	Capacitatea rețelei neurale Demonstrarea faptului că orice funcție continuă poate fi aproximată de o RPM
Boeing Computer Services	SUA	Robert Marcus	Algoritmi neurali pentru parcurgerea rețelei cu cost minim
Boeing High Tech Center	SUA	D. Nguyen	Prelucrare stohastică în RNA
EEG Sytems Lab and Sam Technology Inc., San Francisco	SUA	Alan S. Gevins	Înțelegerea modului de funcționare a creierului uman. Analiza semnalelor biologice cu RNA
E.S.P.C.I., Paris	SUA	L. Personnaz, I. Guyon, G. Dreyfus	Rețele recurente; Aplicații în clasificarea caracterelor
ETL	Japonia	M. Ishikawa	Algoritm de generare a unei structuri neurale skeleton
Fujitsy Ltd.	Japonia	Kazuhiro Matsuo	Rețele auto-organizabile cu învățare competitivă
Hewlett – Packard Labs., Palo Alto	SUA	Egon Loebner	Studiul proceselor de concurență în retina vertebratelor
HFSP	Japonia		Proiectul guvernamental "Științele umane de frontieră" ("Human Frontier Science Program" = HFSP)
Hitachi	Japonia	A. Amano Doya	Metodă de recunoaștere a fonemelor, bazată pe reguli. RNA realizează detecția trăsăturilor acustice Predicție cu RNA Diagnoza leucocitelor
Honeywell CSDD. Golden Valley	SUA	Tarik Samad	Rețele care învață cu algoritmul BKP

Scanned with OKEN Scanner

Tab. A1.1 Proiecte în domeniul reţelelor neurale, în mediul industrial (cont.)

Hughes Aircraft Company, Malibu	SUA	Patrick Castelaz	Reţele celulare (CNN)
Los Alamos National Laboratory	SUA	Geoffrey W. Hoffman, M.W. Benson B.J. Travis Alan Lapedes, R. Farber M. Gremillion	Model neural derivat din analogia realizată între sistemul imunitar şi sistemul nervos central Studiul cortexului auditiv Studiul proprietăţilor reţelei Hopfield Reprezentarea neurală a sistemului vizual al mamiferelor, de la retină la cortexul vizual
Matsushita Electric	Japonia	Sakaue Tamura	Codarea imaginilor Recunoaşterea caracterelor alfanumerice scrise de mână
The MITRE Corporation	SUA	Alexis Wieland	Studiul aspectelor geometrice ale reţelelor neurale. Formularea problemei de clasificare "două spirale"
Nagoya Institute of Technology	Japonia	A. Iwata	Algoritm de compresie a datelor folosind RNA
NASA Ames Research Center	SUA	Pentti Kanerva	Stocarea informaţiei în RNA
National Bureau of Standards, Gaithersburg	SUA	James S. Albus	Modelul CMAC (Cerebral Model Arithmetic Computer)
Naval Ocean Systems Center, San Diego	SUA	Patrick A. Shoemaker	Studiul aplicării metodei Marquardt pentru îmbunătăţirea performanţelor algoritmului BKP
Naval Research Laboratory, Washington	SUA	Harold Szu Arthur D. Fisher, ş.a. Sheldon B. Gardner	Rezolvarea problemelor combinatoriale de optimizare cu RNA Proiectarea arhitecturilor neurale pentru prelucrare paralelă Invăţare entropică: principiul entropiei maxime
NHK Science and Technology Research Laboratories, Tokio	Japonia	Kunihiko Fukushima, Sei Miyake	Modelele cognitron, neocognitron Model neural de atenţie selectivă
NEC Corporation, Kanagawa	Japonia	Kenji Okajima, S. Tanaka, S. Fujuwara K. Yamada	Memorie heteroasociativă cu conexiuni recurente Recunoaşterea caracterelor scrise de mână Identificarea feţelor a 3500 de persoane
Nippon Steel Co.	Japonia		Controlul poluării datorate furnalelor

Scanned with OKEN Scanner

Tab. A1.1 Proiecte în domeniul rețelelor neurale, în mediul industrial (cont.)

NTT	Japonia	Hiramatsu	Utilizarea RNA pentru controlul pachetelor transmise într-o rețea rapidă de telecomunicații
Philips Research Laboratory, Bruxelles	Belgia	Christian J. Wellekins	Recunoașterea automată a semnalului vocal cu RPM
RCA Laboratories, Princeton	SUA	C.H. Anderson	Model neural în care fiecare neuron este stohastic
SAIC, San Diego	SUA	Mark Watson	Modele neurale pentru recunoașterea semnalului vocal
SAM Technology, Inc., San Francisco	SUA	Alan Gevins	Studiul prelucrării semnalului în creierul uman, cu RNA
Schlumberger Palo Alto Research	SUA	Harry Barrow	Studiul câmpurilor receptoare în RNA
Semionics Laboratories, Houston	SUA	S.M. Lamb	Memorie asociativă simplificată, numită "Recognition Memory" (REM)
Tokyo Electric Power Co.	Japonia	Matsuda	Aplicarea rețelei Hofield pentru economisirea energiei electrice
Toshiba	Japonia	M. Arai Hatano	Capacitatea de învățare a rețelelor neurale Recunoașterea caracterelor Kana scrise de mână
TRW, San Diego	SUA	M.H. Myers	Studiul sistemelor dinamice hamiltoniene, cu funcție energie selectabilă; Prima rețea neurală comercială (Mark III)
U.S. Department of Defense (DARPA), Arlington	SUA	Charles Kellum, John A Neff	Raportul DARPA numit "Strategic Computing Initiative" (1988)
VERAC Inc., San Diego	SUA	Bart Kosko	Modelul memoriei asociative bidirecționale. Logică fuzzy, hărți fuzzy
Xerox Corporation, Palo Alto	SUA	B. A. Huberman	Titlul necunoscut

Scanned with OKEN Scanner

Tab. A1.2 Proiecte în domeniul rețelelor neurale, în mediul universitar (1990 – 1996)

Universitatea	Statul	Autori	Problema teoretică abordată
Air Force Institute of Technology (AF U.)	SUA	M. Kabrisky	Implantarea electrozilor semiconductori în creier
U. of Alberta	Canada	M.M. Oguztoreli, T.M. Caelli	Refacerea semnalelor de intrare în RNA, când se cunoaşte numai activitatea câtorva neuroni
U. of Amsterdam	Olanda		Proiect de cercetare la care participă U. din Leiden (cunoaştere), U. din Amsterdam (robotică), U. din Utrecht (imagistică tridimensională) şi U. din Nijmegen (funcţii auditive, control)
Arizona State U.	SUA	Nathan Combs	Reprezentarea internă a informaţiei în RNA
Boston U.	SUA	Stephen Grossberg, Gail Carpenter Michael Cohen	Reţeaua ART Aspecte teoretice legate de reţele paralele
Brown U.	SUA	James A. Anderson Temă doctorat	RNA autoasociative pentru clasificare în domeniul medical Clasificarea ţipetelor sugarilor
Brunel U.	SUA	David Martland	Structuri autoasociative
U. of Bruxelles	Belgia	Luc Steels	Auto – organizare
U. of California, Berkeley	SUA	Morris Hirsch F.H. Eeckman, W. J. Freeman	Dinamica reţelelor competitive Obţinerea funcţiei sigmoide din considerente biologice; Model dinamic al cortexului cerebral
U. of California, Los Angeles	SUA	Josef Skrzypek	Arhitecturi neurale pentru clasificarea texturilor
U. of California, San Diego	SUA	David Rumelhart David Rumelhart, David Zipser M. C. Mozer	RPM Descoperirea trăsăturilor prin învăţare competitivă RPM pentru recunoaşterea formelor invariantă la rotaţie Structuri ontogenice; Proiectul RAMBOT (sistem expert neural) Structuri neurale ierarhice
U. of California, Santa Barbara	SUA	John Moody Terence Smith	Regularizare; structuri neurale ontogenice Utilizarea algoritmilor genetici pentru reţele ontogenice

Scanned with OKEN Scanner

Tab. A1.2 Proiecte în domeniul rețelelor neurale, în mediul universitar (cont.)

California Institute of Technology (Caltech)	SUA	John J. Hopfield Eric Baum A.Yamamura, D. Psaltis	Rețeaua Hopfield Generalizarea algoritmului BKP, prin aplicarea metodei gradientului cu constrângeri Controlere neurale
Carnegie – Mellon U.	SUA	Geoffrey Hinton, David Touretzky Scott Fahlman	Mașina Boltzmann; Aplicarea structurilor neurale pentru prelucrarea informației simbolice Rețeaua Cascor "feedforward" și recurentă
U. of Chicago	SUA	Matthew N. Jensen	Sincronismul în modificarea ponderilor RNA
Clark U.	SUA	David G. Stork	Recunoașterea și sinteza semnalului vocal
Clemson U.	SUA	Edward W. Page, G.A. Tagliarini	Rezolvarea cu RNA a problemelor de optimizare
Cosiglio Nazionale delle Richerche, Pisa	Italia	Sergio Bottini	Codarea în memoriile asociative
Duke U.	SUA	D.F. Starmer	Caracterizarea plasticității sinaptice
Universitat Dusseldorf	Germania	R. Eckmiller	Simularea activității motorii
Ecole Polytechnic of Montreal	Canada	J. Meunier	Modelarea memoriilor asociative pe baza informației biologice
U. of Edinburgh	Marea Britanie	Richard Rohwer	Aspecte temporale ale învățării în RNA
U. of Essex	Marea Britanie	Paul J. Treffner	Model pentru comportament; Aplicarea RNA în imagistică
Harvard U.	SUA	John Daugman	Studiul sistemului vizual
Hebrew U., Ierusalim	Israel	D.J. Amit, H	Analiza modelului Hopfield
Helsinki U. of Technology	Finlanda	Teuvo Kohonen	Rețeaua Kohonen și aplicații în clasificarea fonemelor; cuantizare vectorială supervizată și nesupervizată
Hungarian Academy of Sciences, Budapesta	Ungaria	Peter Erdi, Tamas Roszka	Studiul rețelelor celulare (CNN); Structuri ierarhice

Scanned with OKEN Scanner

Tab. A1.2 Proiecte în domeniul reţelelor neurale, în mediul universitar (cont.)

U. of Illinois, Chicago	SUA	S.Y. Oh	Model pentru memoria asociativă distribuită, pe baza funcţiilor Walsh – Hadamard
Imperial College of Science and Technology, Londra	Marea Britanie	Igor Aleksander	Structuri paralele; Neuron probabilistic
Indiana University, Bloomington	SUA	Eric J. Smythe	Analiza semnalului vocal prin detecţia neurală a tranziţiilor în domeniul frecvenţă
Institute voor Perceptie Ondersoek, Eindhoven	Olanda	D. Beroule	Informaţia în memoria topografică
Instituto di Psicologia del CNR, Roma	Italia	Domenico Parisi, S. Nolfi	Modelarea conexionistă a constrângerilor sinaptice
John Hopkins U., Baltimore	SUA	Terence Sejnowski	Maşina Boltzmann, algoritmul "simulated annealing" Conversia text dactilografiat ⇒ semnal vorbit cu RPM
Keio U.	Japonia	Y. Akiyama	Modelul "maşina Gauss", care unifică modelele McCulloch – Pitts, Hopfield şi Boltzmann
King's College, Londra	Marea Brit.	John Taylor	Proiectarea unui simulator paralel
Kobe U.	Japonia	Kazuyoshi Tsutsumi	Strategii de învăţare pentru rezolvarea problemelor de control din robotică
Koln U.	Germania	K.E. Kurten	Studiul haosului în sistemele neurale recurente
Kyoto U.	Japonia	S. Shinomoto	Introducerea constrângerilor fiziologice în matricea de auto–corelaţie
U. of Maryland	SUA	James A. Reggia	Studiul fenomenului de inhibiţie laterală în sistemul vizual
Massachusetts Institute of Technology (MIT)	SUA	Gerard Y. Vichniac	Construirea funcţiilor Liapunov aditive pentru reţele neurale
U. of Massachusetts	SUA	Michael Arbib, D. Caplan, A.G. Bar	Modele neurolingvistice Abordare a învăţării în RNA, folosind teoria jocurilor
Max Planck Institute, Goettingen	Germania	H. Kuhn, Christoph von der Marlsburg	Aspecte teoretice ale învăţării; structuri asociative

Scanned with OKEN Scanner

Tab. A1.2 Proiecte în domeniul rețelelor neurale, în mediul universitar (cont.)

U. of Minnesota	SUA	M. O. Poliac	Rezolvarea problemelor de alocare a resurselor cu RNA
U. of New Mexico, Las Cruces	SUA	Marcus Cohen	Model neural cu inhibiție laterală; Proiectarea unui calculator optic, care utilizează un rezonator holografic de volum
State U. of New York	SUA	Deborah Walters	Studiul reprezentării variabilelor în rețelele neurale; Aplicații în detectarea contururilor în imagini.
Northeastern U., Boston	SUA	Gail Carpenter Elias Manolakos	Dezvoltarea modelelor ART Studiul aplicării RN în telecomunicații
Osaka U.	Japonia	Mitsuo Kawato	Studiul structurilor neurale ierarhice
U. de Paris	Franța	F. F. Soulie, Le Cun	Evaluarea învățării în structurilor neurale
U. de Paris – Sud	Franța	E. Gelenbe	Aspectul temporal al învățării în RNA
Pontificio Seminario Romano, Roma	Italia	Gianfranco Basti	Științele cunoașterii și importanța filozofică a abordărilor conexioniste
Princeton U.	SUA	Philip Anderson	Modele de spin
Purdue U.	USA	Maneil Tenorio O.K. Ersoy	Segmentarea cu RNA a imaginilor multispectrale, cu zgomot gaussian Prelucrarea și stocarea informației în RNA, folosind transformări
U. of Rochester	SUA	J.A. Feldman David C. Plaut	Modele neurale ale percepției vizuale Model neural pentru recunoașterea vizuală
Roma U.	Italia	G. Parisi	Influența asimetriilor sinaptice, asupra memoriei asociative
Simon Fraser U.	Canada	Diane Ingraham	Studii asupra sistemului vizual artificial
U. of Southern California	SUA	Leon Chua Rama Chellappa	Circuitul lui Chua; Proiectarea circuitului celular universal Refacerea imaginilor cu RNA
U. of Southern Carolina	SUA	Y. Takefuji	Recunoașterea patternurilor în imagini tridimensionale
Stanford U.	SUA	Bernard Widrow Joseph Goodman David Rumelhart	Modelele neurale din anii '50; Sisteme neurale adaptive. Modelele neurale Adaline, Madaline Studiul prelucrării optice Structuri RPM, etc.
Stuttgart Universitat	Germania	W. Banzhaf	Modele ale memoriei

Scanned with OKEN Scanner

Tab. A1.2 Proiecte în domeniul reţelelor neurale, în mediul universitar (cont.)

Syracuse U.	SUA	K.P. Unnikrishnan	Rolul conexiunilor recurente în percepţia vizuală
Technical U. of Denmark, CONNECT	Danemarca	Lars Kai Hansen, Jan Larsen	Teoria învăţării, generalizare, regularizare adaptivă; Aplicaţii în predicţia seriilor de timp, filtrare neliniară, clasificarea imaginilor PET – scan, etc.
Technical U. of Munchen	Germania	K. Schulten	Imbunătăţirea modelelelor artificiale pe baza informaţiei biologice
U. of Texas, Arlington	SUA	Daniel S. Levine	Studiul mecanismelor atenţiei în sistemele neurale de prelucrare a informaţiei
U. of Texas, Austin	SUA	H. Shin, M. Malek	Memoria booleană adresabilă prin conţinut
U. of Tokushima	Japonia	N. Baba	Metodă de aflare a minimului global al funcţiei cost
U. of Tokio	Japonia	Kaoru Nakano Shun–ichi Amari K. Aihara K. Doya	RNA care funcţionează ca un ţesut care se auto–repară Metode matematice pentru caracterizarea RNA Modelul neuronului haotic Modelul "Adaptive Neural Oscillator" (ANO)
U. of Tsukuba	Japonia	Y. Hirai	Modelul HASP
Utah State U.	SUA	G.S. Stiles, D.L. Dean	Funcţionarea memoriei asociative; Transf. inversă M. – Penrose
U of Washington, Seattle	SUA	Arun K. Somani Robert J. Marks II Jenq–Neng Hwang	Structuri neurale compacte Structuri neurale asociative Aplicaţii ale RNA în prelucrarea imaginilor
U. of West Virginia	SUA	Ronald Millechia	Studiul implementărilor analogice ale RN
U. of Wisconsin – Madison	SUA	Charles V. Stewart, C.R. Dyer J. Tompkins,Y. H. Hu	Modele conexioniste pentru vedere stereoscopică Aplicaţii ale RNA în prelucrarea semnalelor EEG, ECG Arhitecturi neurale optimale; Clasificarea semnalelor medicale

Scanned with OKEN Scanner

ANEXA 2

SUMAR AL IMPLEMENTĂRILOR REȚELELOR NEURALE ARTIFICIALE, ÎN MEDIUL UNIVERSITAR ŞI INDUSTRIAL (1990 – 1996)

Un sumar al implementărilor reţelelor neurale, atât în universităţi, cât şi în companii industriale din întreaga lume, este concentrat în tab. A2.1 – A2.5. Ordonarea este realizată după tipul implementării: software sau hardware (analogică / digitală / optică / hibridă).

Numele persoanelor menţionate indică faptul că, acestea au avut contribuţii majore la proiectele amintite, chiar dacă ulterior, aceste persoane au continuat sau nu, să funcţioneze în cadrul universităţilor / companiilor respective.

Sinteza a fost realizată pe baza informaţiilor disponibile în perioada 1990 – 1996. Subliniem faptul că, aceasta este departe de a fi exhaustivă, numărul mare al implementărilor RNA existente făcând dificilă realizarea acestui obiectiv. Pentru detalii privind aplicaţii izolate ale RNA, se pot consulta titlurile incluse în bibliografia prezentei lucrări.

Scanned with OKEN Scanner

Tab. A2.1 Implementări software ale rețelelor neurale, în mediul industrial și universitar (1990 – 1996)

Compania	Statul	Autori	Implementări software
AI Ware Inc., Cleveland	SUA		Software specific aplicației; Pachetul software N–NET 210
ANSPEC (SAIC, San Diego)	SUA		Limbaj de specificare a rețelelor neurale. Specificația este tradusă apoi în limbaj de nivel intermediar, în final compilat
BehavHeuristics	SUA	William R. Hutchinson	"Airline Marketing Tactician"
Bell Communications Research	SUA	D.J. Burr	Cititor de text neural
Bendix Aerospace	SUA	R. Paul Gorman	Clasificarea neurală a țintelor subacvatice cu sonar
U. of California, Los Angeles	SUA	Josef Skrzypek, Eugene Paik	Simulatorul SFINX
U. of California, San Diego	SUA	David Rumelhart, McClelland	Simulatorul PDP
Carnegie – Mellon U.	SUA	Scott Fahlman	Implementarea structurii CASCOR de tip "feedforward" și de tip recurent
U. of Chicago	SUA	Matthew N. Jensen	Sistem interactiv de simulare a RNA (proiectul MacBrain)
U. di Napoli, Napoli	Italia	P.E. Lauria	Assembler pentru o versiune modernă a rețelei "feedforward" deterministe; Simulare pe rețea de Transputere
Helsinki U. of Technology	Finlanda	Teuvo Kohonen	Pachetele software LVQ_PAK1, 1.2, 2 pentru cuantizare vectorială
HNC Inc., San Diego (Hecht – Nielsen Corp.)	SUA	Hecht – Nielsen	Software QuickStrokes pentru recunoașterea neurală a caracterelor scrise de mână; Limbajul AXON; Pachet software NETSET sub Microsoft Windows
Maharishi Intl. U., Fairfield	SUA	Bruce P. Lester	Limbaj pentru rețele neurale, bazat pe Multi – Pascal, (pentru sisteme multiprocesor)
Symbolics, Inc., Burlington	SUA		Plexi: pachet software în Lisp, cu editor grafic inclus
U. di Napoli, Napoli	Italia	Francesco E. Lauria	Assembler pentru rețele neurale
NeuralWare Inc, Pittsburgh	SUA		Software pentru stații SuperStation / PC (calculator gazdă cu procesor 80186 și coprocesor I860) și SuperStation / SPARC

Scanned with OKEN Scanner

Tab. A2.1 Implementări software ale rețelelor neurale, în mediul industrial și universitar (cont.)

Technical U. of Denmark	Danemarca	Lars Kai Hansen, Jan Larsen, Claus Svarer	Simulatorul "Designer Net"
Ward Systems Group	SUA		Pachetul software Neuroshell

Tab. A2.2 Implementări analogice ale rețelelor neurale, în mediul industrial și universitar (1990 – 1996)

Compania	Statul	Autori	Implementări analogice
Adaptive Solutions Inc., Portland	SUA		Circuitul neural CNAPS–1064. Majoritatea algoritmilor neurali pot fi microprogramați în acest circuit
AT & T Laboratories	SUA	E.G. Spenser E.J. Oliver J.S. Denker, D. Schwartz	Utilizarea materialelor cristaline bazate pe oxid de bismut, pentru comutatoare și rezistoare în RNA Calculatorul RELACS, care folosește memorii asociative Implementări neurale VLSI
U of Calgary	Canada	Brian R. Gaines	Implementări neurale VLSI
California Institute of Technology (Caltech)	SUA	Carver Mead, Lazzaro	Cohlea electronică
Clark U.	SUA	David G. Stork, J. Saylor	Rețele neurale analogice pentru probleme de optimizare
Cornell U.	SUA	Clark Mobarry, A. Lewis	Rețele neurale analogice cu elemente programabile
Fujitsu	Japonia		Neurocircuit, folosind tehnologia BiCMOS
J.W. Goethe U., Frankfurt – Main	Germania	D. Tavangarian	Memorie asociativă paralelă, cu acces aleator
Harvard U.	SUA	Babcok, Westervelt	Circuite neurale simple, cu comportare neliniară complexă
Hitachi	Japonia		Rețea neurală cu 540 neuroni și 34 560 sinapse
Hungarian Academy of Sciences, Budapesta	Ungaria	Tamas Roszka	Rețelelor celulare (CNN); Aplicarea circuitului lui Chua și a procesorului universal CNN la prelucrarea imaginilor, etc.
IBM Corporation, Palo Alto	SUA	W. A. Hanson	Rețele asociative paralele analogice, aplicate pentru construirea porților logice elementare, operatori locali și prelucrarea imaginilor

Scanned with OKEN Scanner

Tab. A2.3 Implementări digitale ale rețelelor neurale, în mediul industrial și universitar (1990 – 1996)

Compania	Statul	Autori	Implementări digitale
U. of California, Los Angeles	SUA	James M. Goodwin	Procesor pe baza teoriei spinului
Fujitsu	Japonia		Sistem neural cu 256 circuite PDS Neurocalculator personal NEUROSIM/L. Placa de bază conține un procesor PDS de tip MB86232; funcționează cu 1.5 MCPS în timpul învățării și 4 MCPS în timpul testării (CPS = conexiuni pe secundă)
George Washington U.	SUA	S. Berkovich	Procesor vectorial asociativ: arhitectura este sistolică, concurentă
Imperial College of Science and Technology, Londra	Marea Britanie	Igor Aleksander	Calculatorul neural paralel Wisard II
Intel Corp., Santa Clara	SUA		Circuitul neural 80170NX;
Motorola Inc., Austin	SUA	G. Giles, C. Hunter	Modulul de management al memoriei din progesorul Motorola MC68851 se bazează pe tehnici de testare a memoriilor adresabile prin conținut.
Nagoya Institute of Technology	Japonia		Accelerator neural, cu 4 procesoare PDS de tip MB862220 și 4 porturi cu memorie
NASA	SUA	Robert T. Savely	Tehnologii de implementare a RNA
NEC	Japonia		Neurocalculatorul NeuMan Neurocalculatorul personal NEURO–7, cu 216 000 conexiuni pe secundă, 82 000 neuroni (total de 246 000 conexiuni posibile)
Nippon Steel Corporation	Japonia		Placă neurală pentru calculatoarele personale comerciale NEC. Are la bază 4 – 16 transputere și funcționează cu viteza de 1 MCPS (CPS = conexiuni pe secundă)
SONY	Japonia		Procesor neural paralel, format din 128 procesoare Intel 80860
Texas Instruments, Dallas	SUA	P. Andrew Penz, Guy Hogle	Placă de coprocesor, numită Odyssey, pentru stațiile grafice Explorer, care funcționează ca un coprocesor neural. Odyssey are la bază procesoare TMS32020 și 1 Moctet de memorie

Scanned with OKEN Scanner

Tab. A2.3 Implementări digitale ale rețelelor neurale, în mediul industrial și universitar (1990 – 1996)

Compania	Statul	Autori	Implementări digitale
U. of California, Los Angeles	SUA	James M. Goodwin	Procesor pe baza teoriei spinului
Fujitsu	Japonia		Sistem neural cu 256 circuite PDS Neurocalculator personal NEUROSIM/L. Placa de bază conține un procesor PDS de tip MB86232; funcționează cu 1.5 MCPS în timpul învățării și 4 MCPS în timpul testării (CPS = conexiuni pe secundă)
George Washington U.	SUA	S. Berkovich	Procesor vectorial asociativ: arhitectura este sistolică, concurentă
Imperial College of Science and Technology, Londra	Marea Britanie	Igor Aleksander	Calculatorul neural paralel Wisard II
Intel Corp., Santa Clara	SUA		Circuitul neural 80170NX;
Motorola Inc., Austin	SUA	G. Giles, C. Hunter	Modulul de management al memoriei din progesorul Motorola MC68851 se bazează pe tehnici de testare a memoriilor adresabile prin conținut.
Nagoya Institute of Technology	Japonia		Accelerator neural, cu 4 procesoare PDS de tip MB862220 și 4 porturi cu memorie
NASA	SUA	Robert T. Savely	Tehnologii de implementare a RNA
NEC	Japonia		Neurocalculatorul NeuMan Neurocalculatorul personal NEURO–7, cu 216 000 conexiuni pe secundă, 82 000 neuroni (total de 246 000 conexiuni posibile)
Nippon Steel Corporation	Japonia		Placă neurală pentru calculatoarele personale comerciale NEC. Are la bază 4 – 16 transputere și funcționează cu viteza de 1 MCPS (CPS = conexiuni pe secundă)
SONY	Japonia		Procesor neural paralel, format din 128 procesoare Intel 80860
Texas Instruments, Dallas	SUA	P. Andrew Penz, Guy Hogle	Placă de coprocesor, numită Odyssey, pentru stațiile grafice Explorer, care funcționează ca un coprocesor neural. Odyssey are la bază procesoare TMS32020 și 1 Moctet de memorie

Scanned with OKEN Scanner

Tab. A2.3 Implementări digitale ale reţelelor neurale, în mediul industrial şi universitar (cont.)

Texas Instruments	Marea Britanie	Simon Garth	Circuit neural rapid
TRW, San Diego	SUA	M.H. Myers	Prima reţea neurală comercială (Mark III). Este sistem neural adaptiv de uz general, dotat cu pachet software complet, compatibil cu cel de pe calculatoarele DEC VAX/VMS
U. of Tsukuba	Japonia		Neurocircuit cu 6 neuroni şi 84 sinapse, fiecare dintre acestea cu 64 valori posibile ale ponderilor

Tab. A2.4 Implementări optice ale reţelelelor neurale, în mediul industrial şi universitar (1990 – 1996)

Compania	Statul	Autori	Implementări optice
Air Force of Scientific Research	SUA	C. Lee Giles	Memorii asociative optice
BDM Corporation	SUA	R.A. Athale	Implementare optică a memoriei asociative
U. of California, San Diego	SUA	Clark C. Guest, M. S. Kim	Procesor holografic asociativ; Reţele "Crossbar Associative Networks"; Holografie de volum în cristale fotorefractive;
California Institute of Technology (Caltech)	SUA	Demetri Psaltis	Corelator de imagini electro – optic necorerent, în timp real; Memorie asociativă adresabilă prin conţinut; Holograme bazate pe utilizarea luminii necorerente
Cognitex Corporation	SUA	Steve Richfield	Recunoaşterea optică a caracterelor
U. of Colorado	SUA	Dana Z. Anderson	RNA bazate pe rezonatori optici
Georgia Institute of Technology	SUA	T.K.Gaylord	Implementarea optică a tehnicilor de tip "look – up table"
Hughes Aircraft Company, Malibu	SUA	Bernard H. Soffer	Memorie asociativă holografică

Scanned with OKEN Scanner

Tab. A2.4 Implementări optice ale rețelelelor neurale, în mediul industrial şi universitar (cont.)

Industrial Products Research Inst	Japonia	Nakano	Structura "asociatronul lui Nakano" (memorie optică)
Mitsubishi	Japonia		Neurocircuit optic, în tehnologie GaAs, ce simulează un sistem cu 32 neuroni; Implementarea memoriei asociative Hopfield folosind neurocircuitul menţionat.
U. of New Mexico, Las Cruces	SUA	Marcus Cohen	Calculator optic, cu rezonator holografic de volum Descompunerea imaginilor în domeniul frecvenţă.
NTT	Japonia		RPM optică , cu cristale fotorefractive
Ortel Corporation, Alhambra	SUA	Mark Cronin Golomb	Utilizarea elementelor fotorefractive pentru memoriile asociative optice
U. of Pennsylvania, Philadelphia	SUA	N.H. Farhat	RNA optică 2D, pentru recunoaşterea semnalelor radar
U. of Southern California	SUA	Leon Chua A. Marrakchi, A.Tanguay	Procesorul universal CNN Convertor optic fotorefractiv, coerent – necoerent

Tab. A2.5 Implementări hibride ale rețelelor neurale, în mediul industrial şi universitar (1990 – 1996)

Compania	Statul	Autori	Implementări hibride
Fujitsu	Japonia		Neurocalculator cu arhitectură de tip inel. Conţine 256 de procesoare Texas Instruments TMS320C30, are 8 kocteţi de memorie internă şi 256 kocteţi de memorie externă rapidă
Ministry of International Trade and Industry	Japan		Proiectul guvernamental pentru dispozitive bio – electronice (aplicarea funcţiilor biologice în dispozitivele electronice)
NASA	SUA	Robert T. Savely	Tehnologii de implementare a RNA

Scanned with OKEN Scanner

ANEXA 3

ALTE MODELE DE PERCEPTRON SIMPLU

In multe studii se consideră că, la nivel de micro – structură neurală se află perceptronul simplu cu modelul McCulloch – Pitts, dar având altă funcție de activare decât funcția treaptă. In par. 3.1.1.1 am amintit existența multor alte modele de perceptron simplu, cu şi fără reacție şi considerăm utilă prezentarea succintă a acestora în continuare.

A3.1 Modele de perceptron simplu fără reacție

A3.1.1 Modelul Dahanayake – Upton (Perceptronul simplu "deştept")

In clasa modelelor obținute prin modificarea perceptronului simplu McCulloch – Pitts, în sensul alegerii funcției de activare diferită de funcția treaptă, pot fi incluşi: perceptronul simplu neliniar, perceptronul simplu stohastic, etc., precum şi perceptronul simplu Dahanayake – Upton.

Perceptronul simplu "deştept" propus de Dahanayake şi Upton [D37] se obține înlocuind în modelul McCulloch – Pitts, funcția de activare de tip treaptă cu cea dată de relația (A3.1), pentru care valoarea ieşirii este (A3.2).

$$f(net) = \frac{net}{1 + net^2} \tag{A3.1}$$

$$y = f(net) = f\left(\sum_{i=0}^{M} w_i\ x_i\right) = \frac{\sum_{i=0}^{M} w_i\ x_i}{1 + \left(\sum_{i=0}^{M} w_i\ x_i\right)^2} \tag{A3.2}$$

A3.1.2 Modelul Fukushima

Perceptronul simplu Fukushima este reprezentat în fig. A3.1. S-au notat cu w_i ponderile (pozitive) conexiunilor (numite excitatoare) ce primesc semnalele de intrare x_i, şi cu v_i ponderile (negative) conexiunilor (numite

excitatoare este net_{excit} dată de relaţia (A3.3), iar suma ponderată a celor inhibitoare s-a notat cu net_{inhib} (A3.4). Funcţia de activare *f(net)* are expresia (A3.6), cu *net* din (A3.5), iar valoarea ieşirii perceptronului simplu Fukushima este (A3.7) [C16].

$$net_{excit} = \sum_{i=1}^{M_1} w_i \, x_i \tag{A3.3}$$

$$net_{inhib} = \sum_{i=1}^{M_2} v_i \, q_i \tag{A3.4}$$

$$net = \frac{net_{excit}}{net_{inhib}} - 1 \tag{A3.5}$$

$$f(net) = \begin{cases} net & \text{pentru } net \geq 0 \\ 0 & \text{altfel} \end{cases} \tag{A3.6}$$

$$y = f(net) = f\left(\frac{1 + \sum_{i=1}^{M_1} w_i \, x_i}{1 + \sum_{i=1}^{M_2} v_i \, q_i} - 1 \right) \tag{A3.7}$$

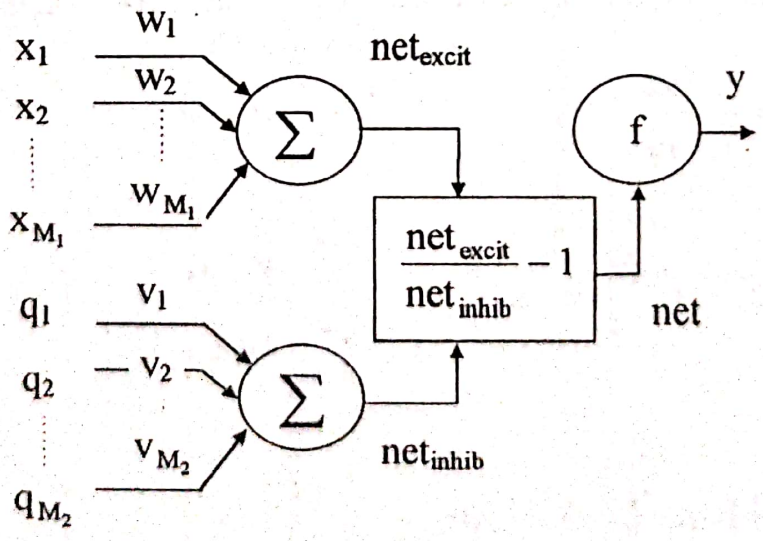


Fig. A3.1 Modelul perceptronului simplu Fukushima

Scanned with OKEN Scanner

A3.1.3 Modelul Lapedes – Farber

Perceptronul simplu Lapedes – Farber are aceeaşi structură cu modelul McCulloch – Pitts. Cu toate acestea, există două deosebiri: în modelul Lapedes–Farber valorile de intrare, valorile *net* şi valoarea de ieşire *y*, sunt dependente de timp (fig. A3.2). De aceea, modelul este dinamic, spre deosebire de cel McCulloch – Pitts, care este un model static. In plus, funcţia de activare *f* este neliniară în modelul Lapedes – Farber [T19].

Valoarea de ieşire a acestuia este:

$$y(t) = f(net(t)) = f\left(\sum_{i=1}^{M} w_i\, x_i(t)\right) \qquad (A3.8)$$

unde $1 \le i \le M$ sunt intrările perceptronului simplu, iar w_i sunt ponderile conexiunilor.

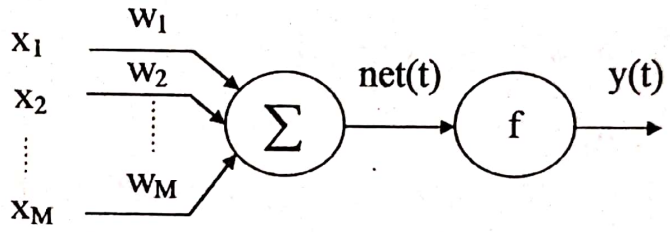


Fig. A3.2 Modelul perceptronului simplu Lapedes – Farber

A3.1.4 Alte modele

Există modele de perceptron simplu care nu se bazează pe calculul sumei ponderate a semnalelor de intrare, ci pe alte operaţii matematice. Printre acestea se pot menţiona modelele de perceptron simplu de tip "high order". Funcţiile de activare (polinomială, "ridge", sigma – pi, etc.) sunt prezentate în anexa 4. Oricare ar fi acestea, modelul perceptronului simplu din această clasă se bazează pe structura prezentată în fig. 3.2.

A3.2 Modele de perceptron simplu cu reacţie

A3.2.1 Modelul Lapedes – Farber

Perceptronul simplu de tip Lapedes – Farber (fig. A3.2) reprezintă un caz particular (în care reacţia este nulă) al perceptronului simplu cu reacţie.

Scanned with OKEN Scanner

A3.2.2 Modelul Back – Tsoi

Back şi Tsoi [T19] au propus o structură de perceptron simplu cu reacţie locală la nivelul sinapsei (fig. 3.5). Fiecare pondere de valoare constantă a fost înlocuită cu o funcţie de transfer liniară, cu poli şi zerouri. Perceptronul simplu utilizat este o versiune modificată a modelului McCulloch – Pitts cu prag:

$$y(t) = f\left(\sum_{i=0}^{M} G_i\left(z^{-1}\right)\, x_i(t) \right) \tag{A3.9}$$

unde $x_i(t)$, *cu* $1 \le i \le M$, sunt valori de intrare, x_0 corespunde intrării de prag (nu s-a reprezentat în figură), iar $G_i\left(z^{-1}\right)$ este funcţia liniară de transfer, cu forma generală:

$$G\left(z^{-1}\right) = \frac{\sum_{j=0}^{n_z} b_j\, z^{-j}}{\sum_{j=0}^{n_p} a_j\, z^{-j}} \tag{A3.10}$$

Valorile a_j, cu $1 \le j \le n_p$ şi b_j, cu $1 \le j \le n_z$, sunt constante, n_p şi n_z sunt numărul polilor şi, respectiv, zerourilor funcţiei G, iar $z^{-j}\, x(t) \overset{\text{def.}}{=} x(t-j)$. Notaţia combinată utilizată, în domeniul timp şi frecvenţă, este cea propusă de lucrarea [T19], scopul fiind compactarea notaţiei şi reliefarea caracteristicilor structurii. Se presupune că a_{n_p}, b_{n_z} sunt nenule.

Valorile de intrare $x_i(t)$, $1 \le i \le M$, pot fi preluate de la ieşirea perceptronului simplu sau, în cazul unei structuri multi – perceptron simplu, pot fi ieşirile unui strat anterior compus din perceptroni simpli. In primul caz, structura are reacţie locală la ieşire, în cel de-al doilea caz are reacţie locală la nivel de sinapsă.

Modelul Lapedes – Farber poate fi considerat ca un caz particular al arhitecturii Back – Tsoi, pentru $a_i = 0$, $1 \le i \le n_p$ şi $a_0 = 1$.

A3.2.3 Modelul Frasconi – Gori – Soda

Perceptronul simplu Frasconi – Gori – Soda este o structură cu reacţie locală la ieşire (fig. 3.6).

Scanned with OKEN Scanner

$$y(t) = f\left(\sum_{j=1}^{m} k_j\, y(t-j) + \sum_{i=1}^{M} w_i\, x_i(t)\right) \qquad \text{(A3.11)}$$

Valorile de intrare x_i, cu $1 \le i \le M$ sunt preluate de la intrările perceptronului simplu sau, în cazul unei structuri multi – perceptron simplu, dintr-un strat anterior compus din perceptroni simpli, k_j sunt constante, ca şi ponderile w_i, cu $1 \le i \le M$, $1 \le j \le m$ (fig. A3.3). Este o generalizare a modelului Lapedes – Farber, în care s-a introdus reacţia locală. S-a notat cu Δ întârzierea unitară [T19].

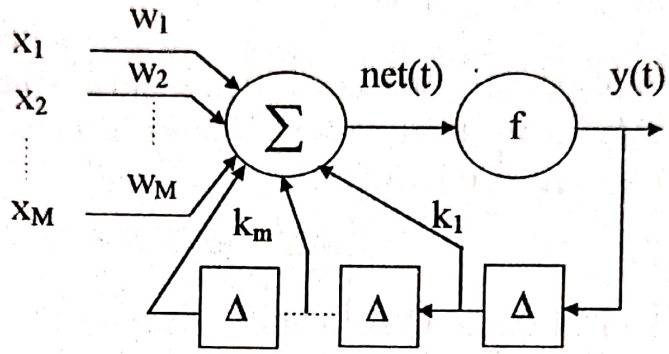


Fig. A3.3 Modelul perceptronului simplu Frasconi – Gori – Soda

A3.2.4 Modelul Frasconi – Gori – Soda generalizat

Tsoi şi Back [T19] au propus generalizarea modelului Frasconi – Gori – Soda, obţinând structura din fig. A3.4. Valoarea de ieşire a modelului este dată de relaţia (A3.12).

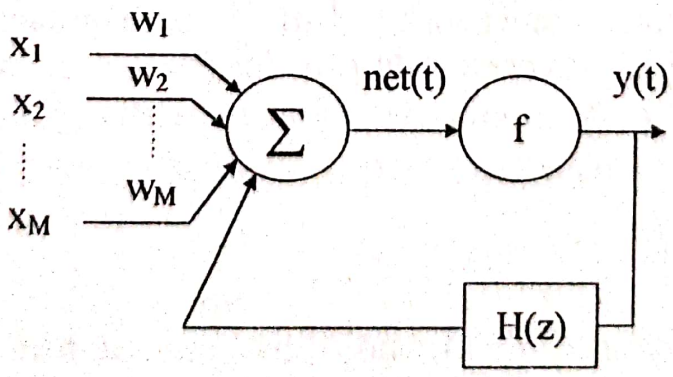


Fig. A3.4 Modelul perceptronului simplu Frasconi – Gori – Soda generalizat

$$y(t) = f\left(\sum_{j=1}^{m} k_j\, y(t-j) + \sum_{j=0}^{n_d} \sum_{i=0}^{M} w_{ij}\, x_i(t-j)\right) \qquad \text{(A3.12)}$$

unde k_j şi w_{ij} sunt constante, $1 \le j \le n_d$, $0 \le i \le M$. S-a notat cu n_d numărul intrărilor întârziate, preluate de la perceptronii simpli din stratul precedent (dacă modelul face parte dintr-o structură multistrat). S-a considerat că intrarea x_0, cu ponderea $(-b_j)$ corespunde pragului perceptronului simplu (nu s-a reprezentat în figură). Funcţia de transfer $H(z)$ poate avea poli şi zerouri.

A3.2.5 Modelul De Vries – Principe

De Vries şi Principe au arătat că se poate modela conexiunea sinaptică printr-un operator de convoluţie. Poate fi considerat ca o generalizare a modelului Frasconi – Gori – Soda [T19].

A3.2.6 Modelul Poddar – Unnikrishnan

Poddar şi Unnikrishnan au propus o clasă de structuri "feedforward" în care fiecare perceptron simplu are memorie, păstrând astfel informaţia referitoare la valorile *net* anterioare [T19].

Valoarea ieşirii modelului este dată de relaţia:

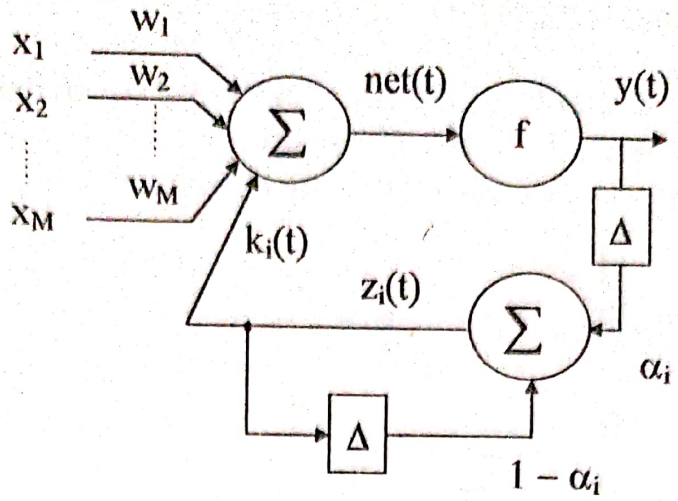
$$y(t) = f\left(\sum_{i=0}^{M} w_i\, x_i(t) + \sum_{i=0}^{M} k_i\, z_i(t)\right) \qquad \text{(A3.13)}$$

unde $z_i(t)$ sunt ieşirile perceptronilor simpli cu memorie (eventual din stratul precedent, într-o structură multistrat), valorile k_i sunt constante, iar $1 \le i \le M$ (fig. A3.5). Indicele $i = 0$ corespunde pragului (nu s-a reprezentate în figură). Perceptronul simplu cu memorie este definit de relaţia (A3.14), cu α_i valoare constantă:

$$z_i(t) = \alpha_i\, y(t-1) + (1-\alpha_i)\, z_i(t-1) \qquad \text{(A3.14)}$$

Arhitectura Poddar – Unnikrishnan este un caz particular al structurii Frasconi – Gori – Soda generalizată, în care funcţia de transfer $H(z)$ are un singur pol. S-a notat cu Δ întârzierea unitară.

Scanned with OKEN Scanner



(0 ≤ i ≤ M, în fig. s-a reprezentat numai unul)

Fig. A3.5 Modelul perceptronului simplu Poddar – Unnikrishnan

A3.2.7 Modelul Dahanayake – Upton

Dahanayake şi Upton [D37] au propus un model cu memorie locală, numit perceptron simplu "inteligent" (fig. A3.6), a cărui valoare de ieşire este dată de relaţia (A3.15), iar funcţia de activare *f*(*net*) este (A3.1).

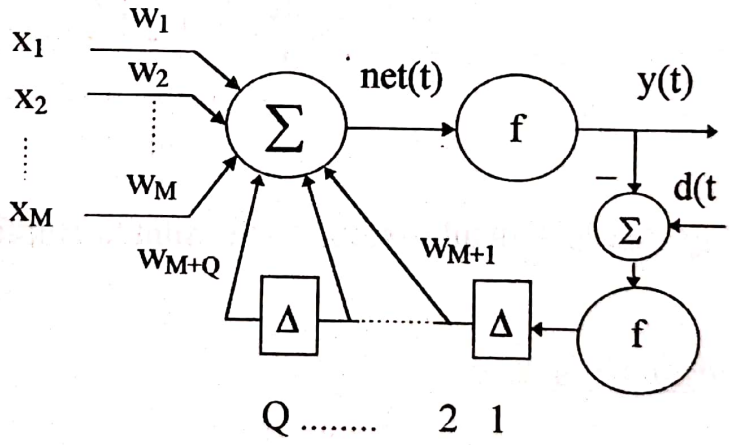


Fig. A3.6 Modelul perceptronului simplu Dahanayake – Upton

$$y(t) = f\left(\sum_{i=0}^{M} w_i\, x_i(t) + \sum_{q=M+1}^{M+Q} w_q\, f\big(d(t-q+M) - y(t-q+M)\big)\right) \quad \text{(A3.15)}$$

S-a notat cu $d(t)$ valoarea ieşirii dorite şi cu Δ întârzierea unitară. Indicele $i = 0$ corespunde, ca şi anterior, pragului (care nu a fost reprezentat în figură).

A3.2.8 Modelul Hopfield

Perceptronul simplu Hopfield (fig. A3.7) este un model dinamic de neuron artificial, care din punct de vedere al structurii este un caz particular al structurii cu reacţie la nivelul activării. Ecuaţiile care descriu modelul sunt:

$$\tau \frac{du}{dt} = -\alpha u + \left(\sum_{i=1}^{M} w_i x_i + b \right) \qquad \text{(A3.16)}$$

$$y = f(u) \qquad \text{(A3.17)}$$

unde x_i sunt valorile semnalului de intrare, u este potenţial de acţiune (activare), τ este o constantă de integrare, α este o constantă. S-au notat cu w_i ponderile şi cu b pragul. Funcţia de activare este de obicei sigmoidală [C16].

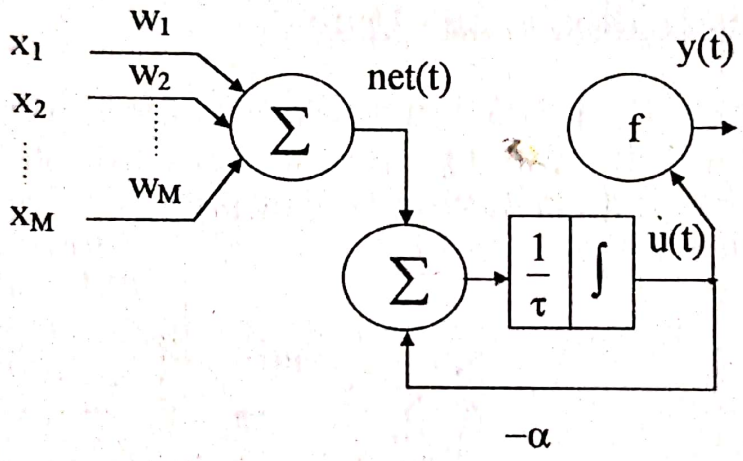


Fig. A3.7 Modelul perceptronului simplu Hopfield

A3.2.9 Modelul Grossberg

Perceptronul simplu Grossberg este, de asemenea, un model dinamic ce reprezintă caz particular al structurii cu reacţie la nivelul activării (fig. A3.8). Este descris de ecuaţia stării perceptronului simplu (A3.18) şi de legea de învăţare. Constanta γ se numeşte constantă de normalizare a activării, iar β este o constantă pentru control. Funcţia de activare f este de obicei neliniară şi $y = f(u)$ [C16].

$$\tau \frac{du}{dt} = -\alpha u + (\gamma - \beta u)\left[\sum_{i=1}^{M} w_i f_i(u_i) + b \right] \qquad \text{(A3.18)}$$

Scanned with OKEN Scanner

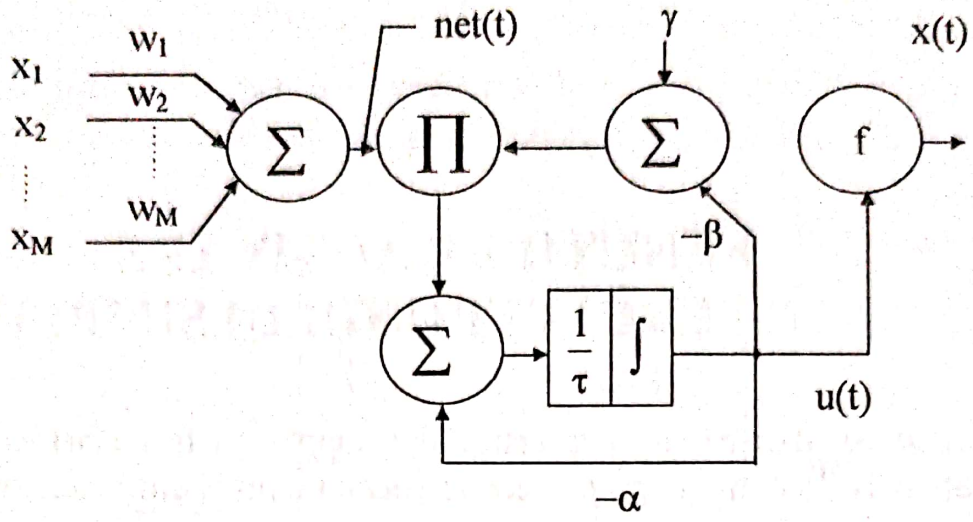


Fig. A3.8 Modelul perceptronului simplu Grossberg

Scanned with OKEN Scanner

ANEXA 4

FUNCȚII DE ACTIVARE ALE PERCEPTRONULUI SIMPLU

Funcțiile de activare ale perceptronului simplu au fost clasificate în par. 3.1.1.2 (tab. 3.2). In continuare, prezentăm succint o parte dintre acestea.

A4.1 Funcții de activare de tip prag

Funcțiile de activare de tip prag (unipolar sau bipolar) sunt definite de relațiile (A4.1), (A4.2) şi reprezentate grafic în fig. A4.1.

A4.1.1 Prag unipolar

$$f(\text{net}) = \begin{cases} 1 & \text{pentru } \text{net} \geq 0 \\ 0 & \text{pentru } \text{net} < 0 \end{cases} \qquad \text{(A4.1)}$$

A4.1.2 Prag bipolar

$$f(\text{net}) = \begin{cases} 1 & \text{pentru } \text{net} \geq 0 \\ -1 & \text{pentru } \text{net} < 0 \end{cases} \qquad \text{(A4.2)}$$

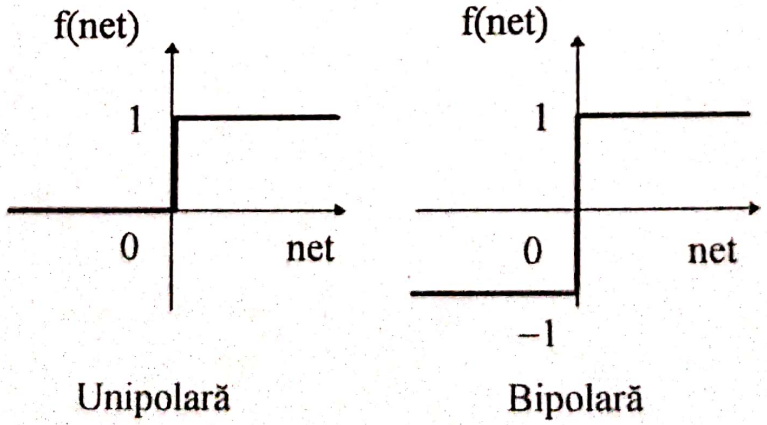
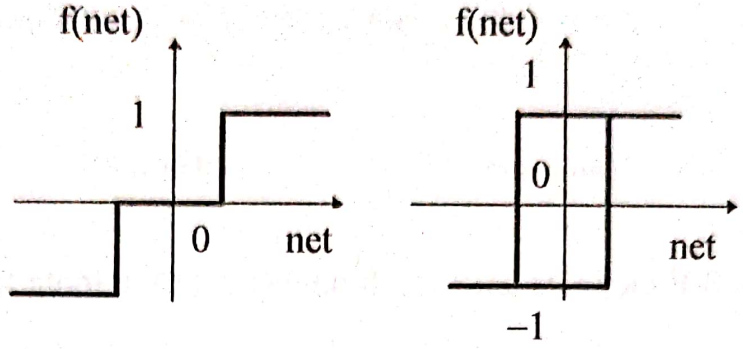


Fig. A4.1 Funcția treaptă (prag) "Heaviside" unipolară şi bipolară

A4.1.3 Alte funcţii prag

Alte alegeri posibile pentru funcţia prag sunt: funcţia prag "cu zonă moartă" şi funcţia prag cu histerezis, ambele reprezentate în fig. A4.2 [C16].



Prag "cu zonă moartă" Prag cu histerezis

Fig. A4.2 Funcţia prag "cu zonă moartă" şi funcţia prag cu histerezis

A4.2 Funcţii de activare liniare

Funcţiile de activare liniare (de tip rampă) unipolare sau bipolare sunt definite de relaţiile (A4.3), (A4.4) şi reprezentate grafic în fig. A4.3.

A4.2.1 Rampă unipolară

$$f(net) = \begin{cases} 0 & \text{pentru net} < -\dfrac{1}{s} \\ s\ net & \text{pentru } -\dfrac{1}{s} \le net \le \dfrac{1}{s} \\ 1 & \text{pentru net} > \dfrac{1}{s} \end{cases} \qquad (A4.3)$$

A4.2.2 Rampă bipolară

$$f(net) = \begin{cases} -1 & \text{pentru net} < -\dfrac{1}{s} \\ s\ net & \text{pentru } -\dfrac{1}{s} \le net \le \dfrac{1}{s} \\ 1 & \text{pentru net} > \dfrac{1}{s} \end{cases} \qquad (A4.4)$$

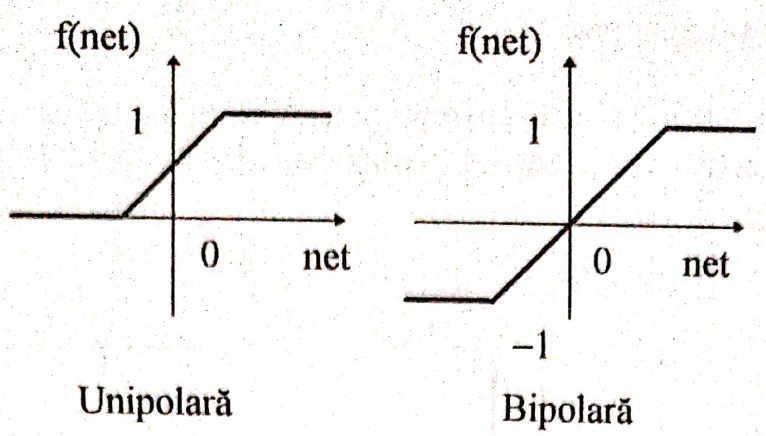


Fig. A4.3 Funcţia de activare liniară (rampă) unipolară şi bipolară

A4.2.3 Rampă cu zonă moartă

Funcţia rampă "cu zonă moartă" este reprezentată în fig. A4.4 [C16].

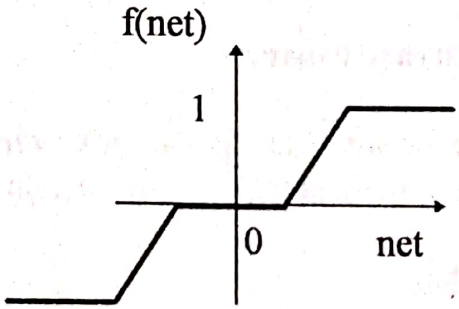


Fig. A4.4 Funcţie de activare rampă "cu zonă moartă"

A4.2.4 Funcţie modul

Funcţia modul este definită de (A4.5) şi reprezentată în fig. A4.5.

$$f(net) = |\, net \,| \qquad (A4.5)$$

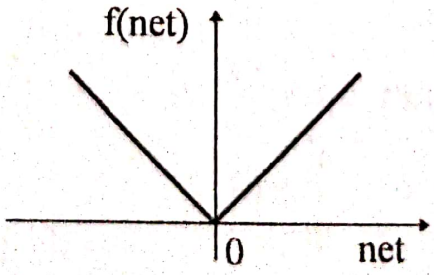


Fig. A4.5 Funcţia de activare modul

A4.2.5 Funcţie triunghiulară

Funcţia de activare poate fi aleasă de tip "fereastră" (funcţie nemonotonă) liniară (fig. A4.6). Suprapunerea unor astfel de funcţii pentru cazul structurilor formate din mai mulţi perceptroni simpli are drept efect activarea mai multor noduri, dar a fiecăruia într-o măsură diferită, pentru aceeaşi valoare de intrare [H11].

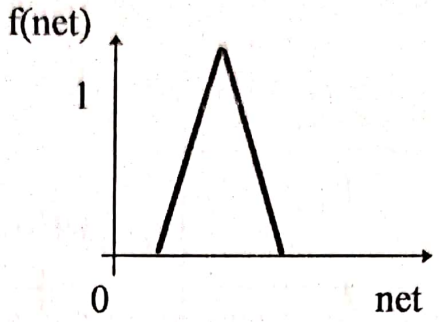


Fig. A4.6 Funcţia de activare triunghiulară

A4.2.6 Funcţie trapezoidală

Debenham şi Garth [L21] au propus utilizarea unei funcţii de activare (fig. A4.7) trapezoidale pentru perceptronii simpli componenţi ai unei structuri "feedforward".

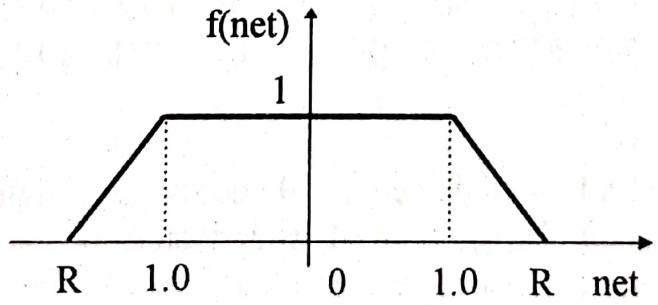


Fig. A4.7 Funcţia de activare Debenham – Garth

A4.3 Funcţii de activare neliniare

A4.3.1 Funcţie sigmoidală monotonă

Funcţia sigmoidală monotonă unipolară a fost definită de relaţia (3.8), iar cea bipolară de (3.9), unde s-a notat cu *s* panta funcţiei. Reprezentarea grafică a

funcţiei şi a derivatei sale, atât în cazul unipolar, cât şi în cazul bipolar este realizată în fig. A4.8.

Alte alegeri posibile de funcţii de activare sigmoidale sunt (3.10) sau (3.11) [F18]. Si şi Michel [S66] au propus utilizarea unei succesiuni de funcţii sigmoide, care să aproximeze pas cu pas, o funcţie treaptă compusă din mai multe segmente, similară celei obţinute prin reprezentarea caracteristicii de cuantizare a unui sistem.

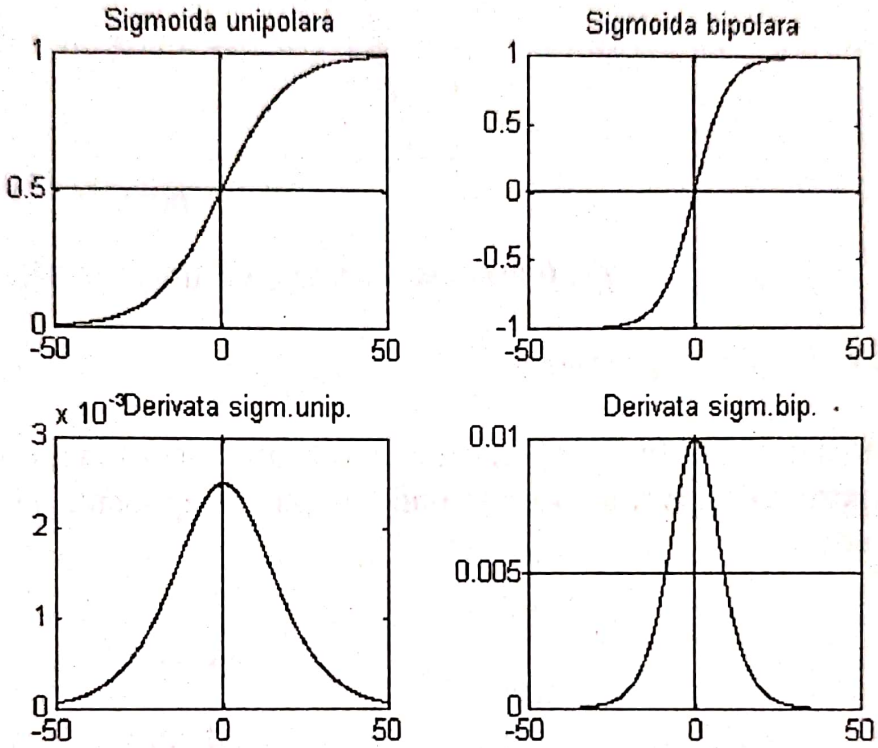


Fig. A4.8 Funcţii de activare de tip tangentă sigmoidală (unipolară şi bipolară) şi derivatele lor

A4.3.2 Funcţie sigmoidală nemonotonă

Dahanayake şi Upton au propus utilizarea funcţiei de activare (A3.1). Funcţia de activare şi derivata sa sunt prezentate în fig. A4.9. Se observă că, funcţia de activare este nemonotonă, cu valori în intervalul [–0.5, +0.5]. Pentru argument finit, derivata sa se anulează de două ori şi are atât valori pozitive, cât şi negative.

Scanned with OKEN Scanner

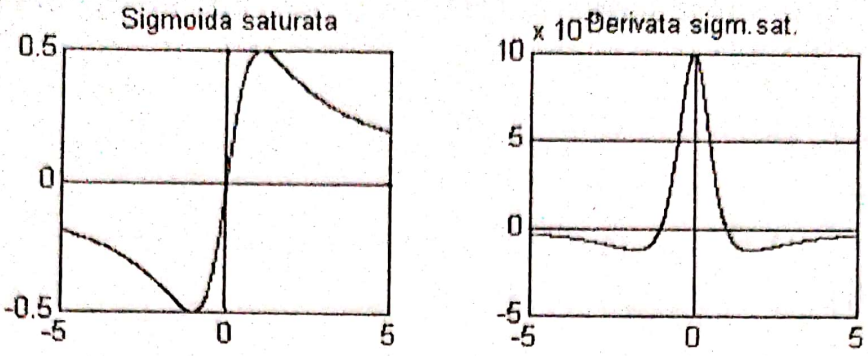


Fig. A4.9 Funcţia de activare sigmoidă nemonotonă

A4.3.3 Polinomială trigonometrică

Funcţia de activare a perceptronului simplu se poate construi pe baza unor relaţii cunoscute din teoria clasică a aproximării funcţiilor. Astfel, s-a sugerat utilizarea funcţiei de activare [K8]:

$$f(net) = \sin(net) \tag{A4.6}$$

A4.3.4 Funcţie cu bază radială

Perceptronii simpli cu funcţie de activare cu bază radială (RBF— "Radial Basis Function") au o valoare a ieşirii dată de relaţia (A4.7), unde $\varphi(r)$ este o funcţie nucleu aleasă, $\{x^{(c)}\}$ sunt centrii (aleşi) ale funcţiei, în număr de M [W27, K9].

$$f(\mathbf{x}) = \varphi\left(\left\|\mathbf{x} - \mathbf{x}_i^{(c)}\right\|\right) \tag{A4.7}$$

Moody şi Darken [M503] au studiat modalităţi de stabilire adaptivă a centrilor pentru funcţiile RBF, propunând metode performante pentru învăţarea în reţele compuse din astfel de perceptroni simpli. Poggio şi Girosi [P200] au propus un formalism riguros pentru reţele RBF şi determinarea adaptivă a centrilor [M17].

Printre funcţiile $\varphi(r)$ utilizate, se numără [W27]:

❶ Funcţie $\varphi(r)$ gaussiană

Funcţia $\varphi(r)$ gaussiană este dată de relaţia (A4.8). Funcţia de activare RBF a perceptronului este (A4.9) şi este reprezentată grafic în fig. A4.10 [W27].

Scanned with OKEN Scanner

S-au notat cu σ_i deviaţia standard, care, ca şi valoarea medie, se modifică în timpul învăţării [E5, M17].

$$\varphi(r) = \exp(-r^2/2) \tag{A4.8}$$

$$f(x) = \exp\left(-\frac{\sum_{i=1}^{M} \left\| x_i - x_i^{(c)} \right\|^2}{2\sigma_i^2}\right) \tag{A4.9}$$

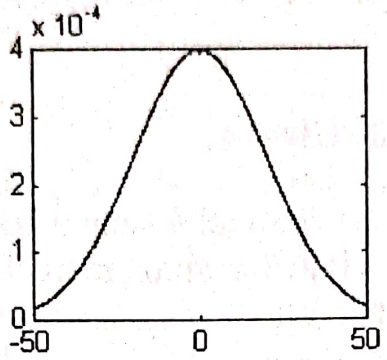


Fig. A4.10 Funcţia de activare gaussiană

❷ Funcţie $\varphi(r)$ "pălărie mexicană"

Funcţia $\varphi(r)$ "pălărie mexicană" este reprezentată în fig. A4.11 [B9, L21].

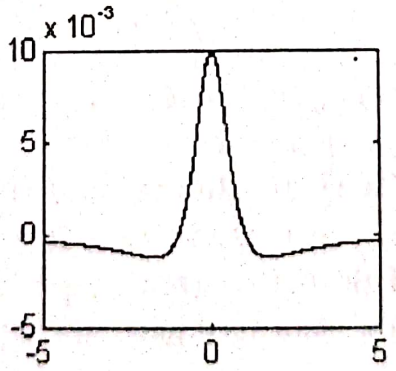


Fig. A4.11 Funcţia $\varphi(r)$ "pălărie mexicană"

❸ Alte funcţii $\varphi(r)$

Printre alte posibilităţi de alegere a funcţiei $\varphi(r)$ se numără [W27]:

$$\varphi(r) = r \tag{A4.10}$$

$$\varphi(r) = r^3 \tag{A4.11}$$

$$\varphi(r) = r^2 \log r \tag{A4.12}$$

$$\varphi(r) = \left(r^2 + c^2\right)^{1/2} \tag{A4.13}$$

$$\varphi(r) = \left(r^2 + c^2\right)^{-1/2} \tag{A4.14}$$

A4.3.5 Funcţii de activare de ordin superior

Din clasa funcţiilor de activare de ordin superior (HONN — "High Order Neural Networks") fac parte funcţiile "ridge", funcţia sigma – pi, funcţiile spline, funcţiile polinomiale algebrice, etc.

❶ Funcţie de activare "ridge"

Un nod cu funcţie de activare "ridge" (A4.15) este de fapt o reţea pi – sigma (PSN — "Pi–Sigma Network") [K33, W27], unde *s* este ordinul acesteia.

$$f(\mathbf{x}) = \prod_{i=1}^{s} \left(\mathbf{w}_i^T \mathbf{x} + b_i\right) \tag{A4.15}$$

❷ Funcţie de activare sigma – pi

Funcţia de activare sigma – pi, definită de (A4.16), este denumită astfel datorită celor două operaţii aritmetice ce o caracterizează: suma şi produsul. Spre deosebire de alte legi de activare, suma ponderată a intrărilor este înlocuită cu suma ponderată a produselor valorilor de intrare. Perceptronul simplu caracterizat de această funcţie de activare se numeşte perceptron simplu sigma – pi [K8, R3, H47].

$$y = f\left(\sum_{i=1}^{M} w_i \prod_{k} x_{i_k} + b\right) \tag{A4.16}$$

Scanned with OKEN Scanner

❸ Funcţie polinomială algebrică (de grad $s-1$)

Funcţia de activare polinomială algebrică se obţine alegând în expresia (A4.17) a valorii de ieşire a perceptronului simplu funcţia $\varphi(x)$ de forma (A4.18), unde $\{\varphi_{i-1}\}$ este un set de polinoame ortogonale [W27].

$$y = f\left(\sum_{i=1}^{M} c_i \varphi_{i-1}(x)\right) \tag{A4.17}$$

$$\varphi_{i-1}(x) = x^{i-1} \tag{A4.18}$$

❹ Funcţie de activare de tip "projection pursuit"

Funcţia de activare de tip "projection pursuit" este o combinaţie liniară de funcţii Hermite [K33]:

$$f(x) = \sum_{r=1}^{s} c_r h_r(x) \tag{A4.19}$$

unde $h_r(x)$ sunt funcţii ortonormale Hermite şi s este ordinul funcţiei de activare.

A4.3.6 Alte funcţii de activare

Adams şi Bye [A5] au propus funcţia de activare (A4.20). Aceasta selectează zone neliniare în planul bidimensional, respectiv volume neliniare în spaţiul multidimensional.

$$f(net) = \frac{1}{1+\alpha d} \tag{A4.20}$$

Dintre cazurile particulare amintim:

❶ $$d = \sum_{i=1}^{M} (w_i - x_i)^2 \tag{A4.21}$$

Pentru valori pozitive ale parametrului α, valoarea ieşirii perceptronului simplu aparţine intervalului [0, 1]. Tot acest parametru determină aria zonei circulare selectate în planul bidimensional, în cazul perceptronului simplu cu două intrări.

Scanned with OKEN Scanner

❷ $$d = k_1 (w_1 - x_1)^2 + k_2 (w_2 - x_2)^2 \qquad \text{(A4.22)}$$

Parametrul α este unitar. Raportul dintre k_1 şi k_2 (valori pozitive) determină forma zonei eliptice aliniată cu axele, selectată în planul bidimensional [A5].

❸ $$d = [k_1 k_3 + k_2(1 - k_3)] (w_1 - x_1)^2 + [k_2 k_3 + k_1(1 - k_3)] (w_2 - x2)^2 -$$

$$- (k_2 - k_1) \left[\sqrt{k_3} (1 - k_3)\right](w_1 - x_1) (w_2 - x_2) \qquad \text{(A4.23)}$$

care este de fapt o expresie de forma (A4.24), cu c_1, c_2, c_3, constante.

$$c_1 (w_1 - x_1)^2 + c_2(w_2 - x_2)^2 - c_3 (w_1 - x_1) (w_2 - x_2) \qquad \text{(A4.24)}$$

Parametrul α este unitar. Raportul dintre valorile pozitive k_1 şi k_2 determină forma zonei eliptice (rotită cu un unghi θ în raport cu axele). Valoarea parametrului k_3 este $k_3 = cos^2\theta$), selectată în planul bidimensional [A5].

Modelul perceptronului simplu obţinut poate fi considerat ca o generalizare a celui cu funcţie de activare RBF gaussiană.

A4.4 Funcţie de activare stohastică

Alegând $s = T^{-1}$ în relaţia (3.8), cu parametrul T numit temperatură, se obţine:

$$f(net) = \frac{1}{1 + \exp(-net/T)} \qquad \text{(A4.25)}$$

Intrucât domeniul valorilor funcţiei este [0, +1], $f(net)$ poate fi considerată probabilitatea p ca valoarea de ieşire y a perceptronului simplu să ia valoarea +1. Modelul obţinut se numeşte perceptron simplu stohastic [K8, S36].

$$p(y \leftarrow 1) = \frac{1}{1 + \exp(-net\ /\ T)} \qquad \text{(A4.26)}$$

Trebuie precizat faptul că, alegerea funcţiei de activare dintre cele enumerate anterior, este dependentă atât de algoritmul de învăţare al modelului neural considerat, cât şi de implementarea finală dorită (software sau hardware). Precizarea acestor aspecte depăşeşte cadrul prezentei anexe.

Scanned with OKEN Scanner

ANEXA 5

REZULTATELE STUDIULUI PRIVIND MODIFICAREA ADAPTIVĂ A PANTEI FUNCȚIEI DE ACTIVARE

Prezentăm în continuare, rezultate ale experimentului 9 (par. 6.4.3.2), referitoare la influența termenului negativ γ de învățare a pantei funcției de activare [D204, L205], funcție dată de relația (3.9) (multiplicată cu o constantă) și notată $f_1(s, net)$, sau de relația (3.11) și notată $f_2(s, net)$. Funcțiile cost considerate sunt estimatori robuști de tip M (tab. 6.1).

Rețeaua a învățat 90 de epoci (un număr total de 51 840 patternuri de intrare). Ponderile și pantele inițiale au fost alese aceleași ca în experimentele din cap. 6. Valorile parametrilor, pantele minimă și maximă și eroarea de test în sens L_2 sunt prezentate în tab. A5.1, iar curbele de învățare în fig. A5.1 – A5.6.

Pentru funcția de activare $f_1(s, net)$, ierarhia performanțelor este: ① norma L_2, ② funcția cost logistică, ③ funcția Hampel, cu rezultatele cele mai bune pentru criteriul L_2.

Pentru $f_2(s, net)$, ierarhia se modifică: ① funcția cost logistică, ② Hampel, ③ norma L_2, cu performanțele cele mai bune pentru funcția logistică. De asemenea, aceasta este mai stabilă la variațiile parametrului γ (tab. A5.1, A5.2).

Eroarea pătratică medie de test, după 90 de epoci, atât pentru $\gamma < 0$ cât și pentru $\gamma > 0$, este prezentată în tab. A5.2 (parametrii constanți sunt aceeași ca în tab. A5.1). Tendința crescătoare sau descrescătoare a erorii de test, în funcție de parametrul γ, a fost marcată cu săgeți. In aproape toate cazurile considerate, (cu excepția ultimei linii a tabelului), alegerea unor valori negative pentru parametrul γ îmbunătățește rezultatele în faza de test, până la un moment dat când, încep să apară oscilații (de exemplu, în fig. A5.7, după ce rețeaua a învățat timp de câteva sute de epoci, începe un proces de oscilație). Momentul de oprire al descreșterii valorii parametrului γ trebuie stabilit cu precizie.

A5.1 Strategia de învățare "flip – flop"

Pe baza rezultatelor anterioare, a fost propusă strategia de învățare "flip–flop" [D204]. Rețeaua învață timp de 90 de epoci cu un termen de învățare a pantei $\gamma < 0$, apoi următoarele 90 de epoci cu $\gamma > 0$ ($|\gamma|$ același).

Scanned with OKEN Scanner

Tab. A5.1 Influenţa termenului γ, de învăţare a pantei, asupra erorii de test, în sens L_2

90 epoci, $\alpha = 0.1$, $\rho = 0.0001$, $s_{min} = 0.3$, $s_{max} = 1.5$,
semnal dorit = semnal PE corect, semnal de test = semnal PE cu zgomot

Funcţia cost	Funcţia de activare	γ	Eroarea de test minimă (L_2)
Norma L_2	$f_1(s, net)$	–0.00001	0.064614
		–0.0001	0.064573
		–0.001	osc.*
		–0.01	osc.
	$f_2(s, net)$	–0.001	osc.
		–0.01	osc.
		–0.1	0.071082
Hampel ($\beta = 0.5$)	$f_1(s, net)$	–0.00001	0.072077
		–0.0001	0.070824
		–0.001	osc.
		–0.01	osc.
	$f_2(s, net)$	–0.00001	0.070453
		–0.0001	0.070624
		–0.001	osc.
		–0.01	osc.
Logistică ($\beta = 0.1$)	$f_1(s, net)$	–0.00001	0.068117
		–0.0001	0.068271
		–0.001	0.06829
	$f_2(s, net)$	–0.00001	0.068680
		–0.001	0.068789
		–0.1	0.068873

*osc. = oscilaţii

Eroarea de test a fost comparată cu cea obţinută pentru cazul învăţării continue timp de 180 de epoci cu $\gamma > 0$. Condiţiile iniţiale au fost cele din tab. A5.1, cu excepţia numărului de epoci. Rezultatele sunt prezentate în tabelul A5.2. In situaţiile marcate cu semnul * (tab. A5.2), strategia "flip – flop" a condus la rezultate mai bune decât învăţarea standard (cu valoare de acelaşi semn a parametrului γ). Rezultatele indicate au fost obţinute pentru funcţia de activare $f_2(s, net)$, pentru ambele valori γ luate în consideraţie.

Scanned with OKEN Scanner

Pentru cazul normei L_2 a erorii şi $f_1(s, net)$ (ambele valori ale parametrului γ), eroarea de test după 90 de epoci ($\gamma < 0$) este mai scăzută decât cea obţinută după încă 90 de epoci cu $\gamma > 0$. De asemenea, este mai mică decât eroarea de test după un număr total de 180 de epoci cu aceeaşi valoare $\gamma > 0$.

Studiind curbele de învăţare pentru aceste cazuri (fig. A5.9), se observă că, după aproximativ 90 de epoci are loc o creştere a erorii. Curbele de învăţare (fig. A5.9 – A5.14) nu pun în evidenţă diferenţe semnificative între strategia de învăţare standard şi cea "flip – flop", cu excepţia fig. A5.14. Aceasta se remarcă prin descreşterea mai accentuată a erorii pe parcursul primelor epoci (cu aproximativ 25% mai rapidă).

Valorile PSNR obţinute pentru $\gamma < 0$, pentru toate funcţiile cost şi funcţiile de activare considerate nu variază semnificativ, de aceea nu se poate afirma că, faţă de cazul anterior, performanţele se îmbunătăţesc [D205]. Are loc însă o creştere a vitezei de convergenţă şi, de asemenea, tehnica "flip – flop" ajută structura neurală să depăşească, prin comutarea periodică a semnului parametrului γ, eventualele puncte de minim local.

Notaţiile folosite se referă la : a) "flip – flop" cu $\gamma = 0.00001$, b) strategie standard cu $\gamma = 0.00001$, c) "flip – flop" cu $\gamma = 0.0001$ şi d) strategie standard cu $\gamma = 0.0001$.

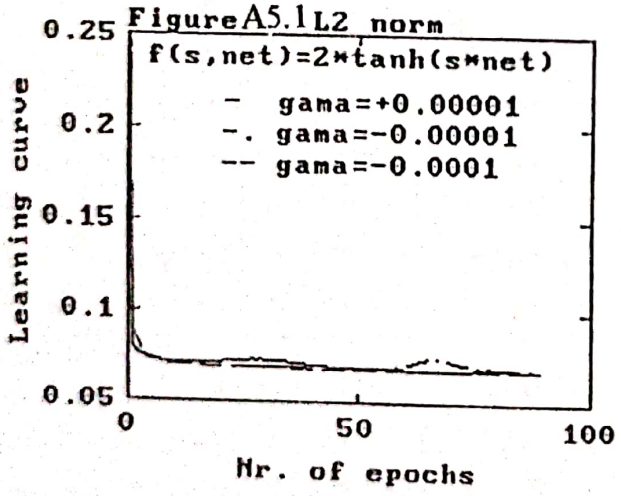


Fig. A5.1 Curbele de învăţare (eroare de învăţare ca funcţie de numărul de epoci) pentru norma L_2, diferite valori ale param. γ şi funcţia de activare $f_1(s, net)$

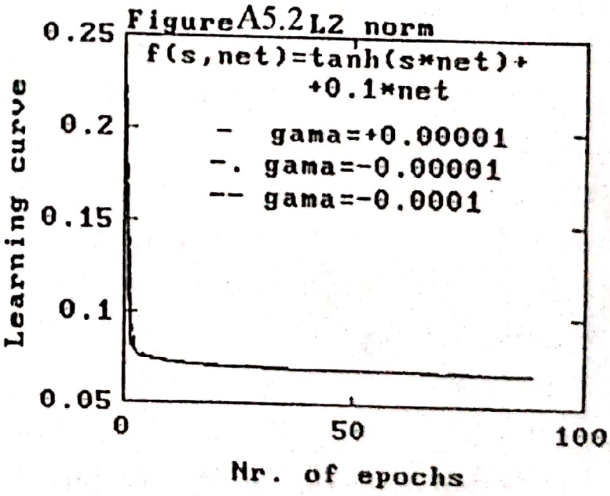


Fig. A5.2 Curbele de învăţare pentru norma L_2, diferite valori ale param. γ şi funcţia de activare $f_2(s, net)$

Scanned with OKEN Scanner

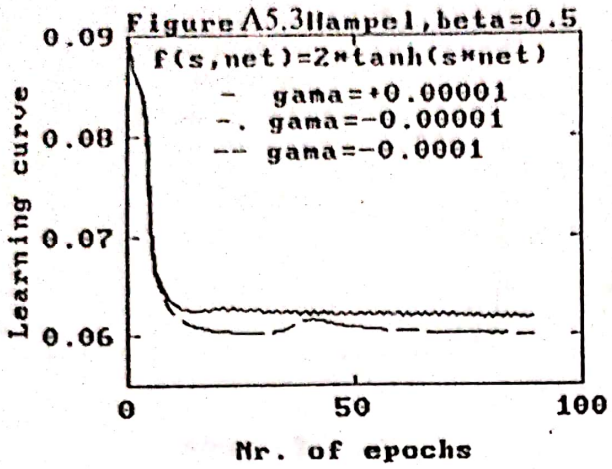


Fig. A5.3 Curbele de învăţare pentru funcţia cost Hampel, $\beta = 0.5$ şi funcţia de activare $f_1(s, net)$

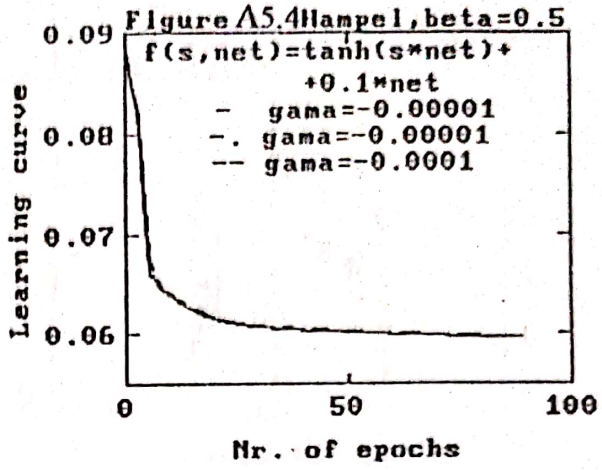


Fig. A5.4 Curbele de învăţare pentru funcţia Hampel, $\beta = 0.5$ şi funcţia de activare $f_2(s, net)$

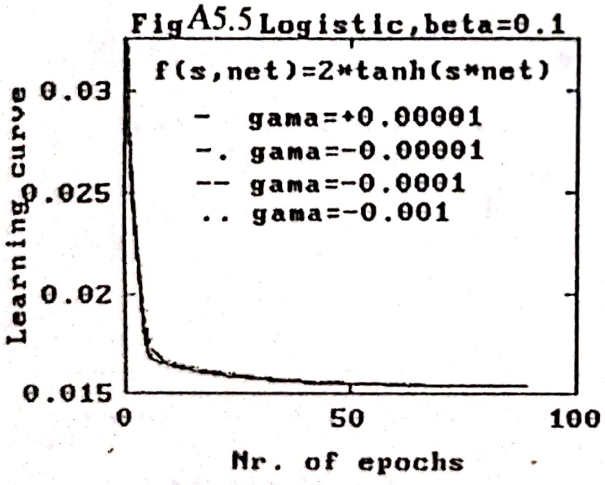


Fig. A5.5 Curbele de învăţare pentru funcţia cost logistică, $\beta = 0.1$ şi funcţia de activare $f_1(s, net)$

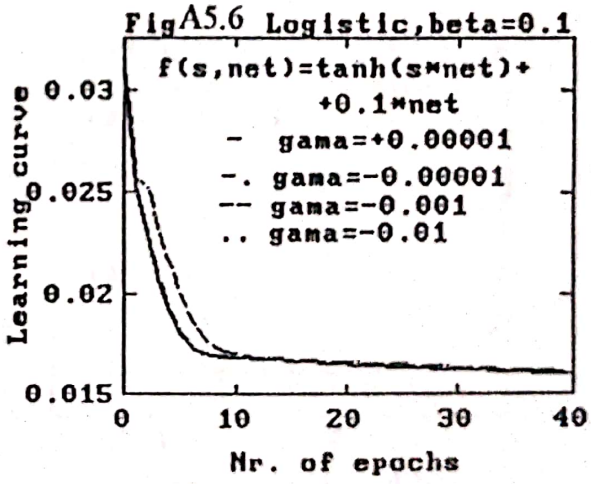


Fig. A5.6 Curbele de învăţare pentru funcţia cost logistică, $\beta = 0.1$ şi funcţia de activare $f_2(s, net)$

Scanned with OKEN Scanner

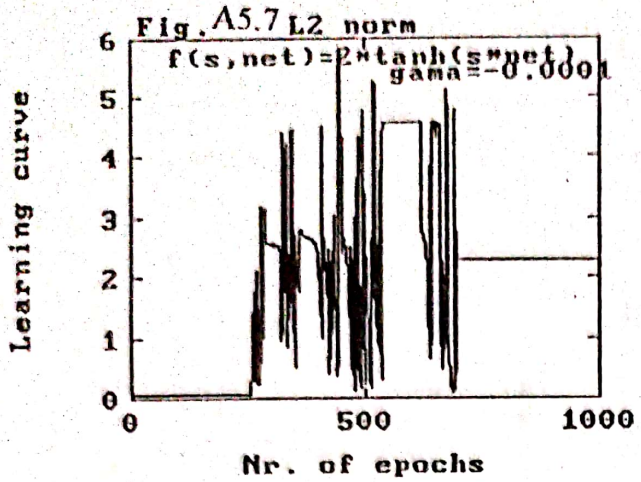


Fig. A5.7 Curbele de învăţare pentru norma L_2, γ=–0.0001 şi funcţia de activare $f_1(s, net)$

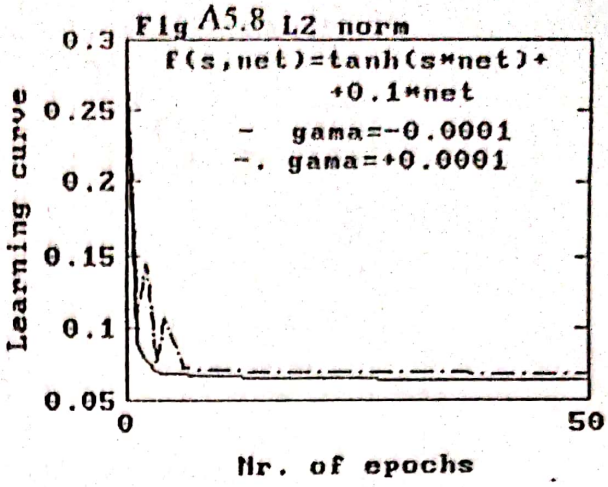


Fig. A5.8 Curbele de învăţare pentru norma L_2, valori negative ale γ şi funcţia de activare $f_2(s, net)$

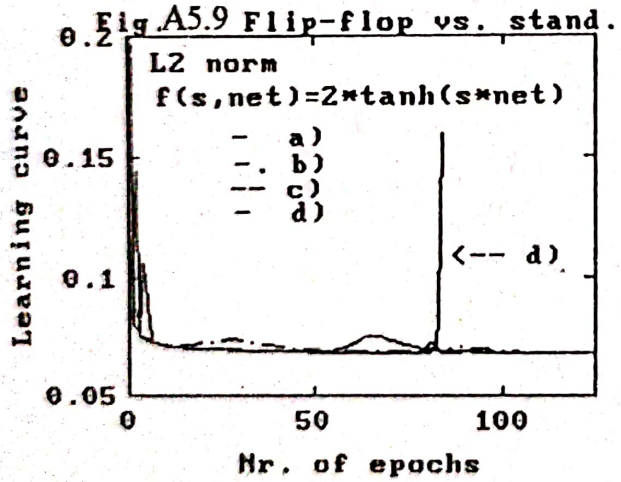


Fig. A5.9 Curbele de învăţare pentru strategiile "flip–flop" şi standard, norma L_2, diferite valori ale parametrului γ şi funcţia de activare $f_1(s, net)$

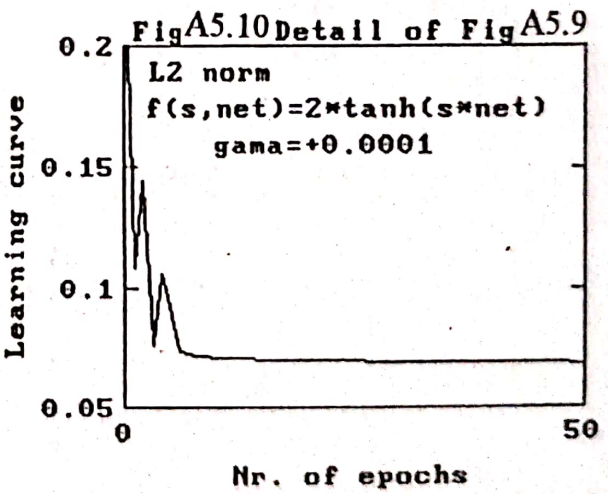


Fig. A5.10 Detaliu al fig. A5.9, pentru $\gamma = 0.0001$

Scanned with OKEN Scanner

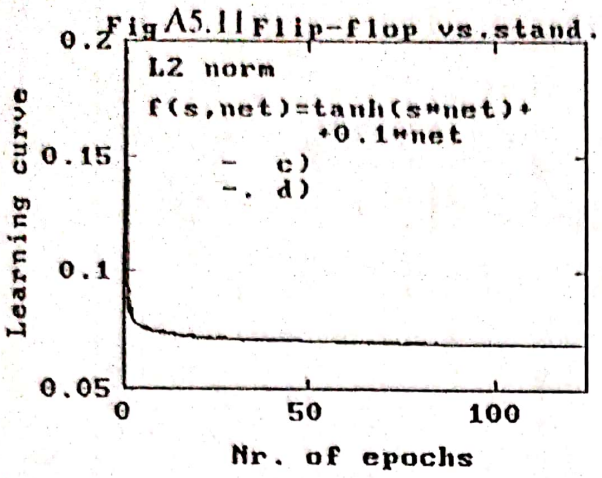


Fig. A5.11 Curbele de învăţare pentru strategiile "flip–flop" şi standard, norma L_2, diferite valori ale parametrului γ şi funcţia de activare $f_2(s, net)$

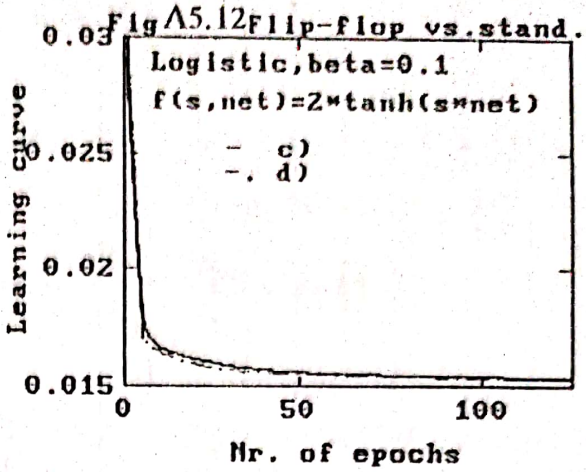


Fig. A5.12 Curbele de învăţare pentru strategiile "flip–flop" şi standard, funcţia cost logistică, $\beta = 0.1$, diferite valori ale param γ şi funcţia de activare $f_1(s, net)$

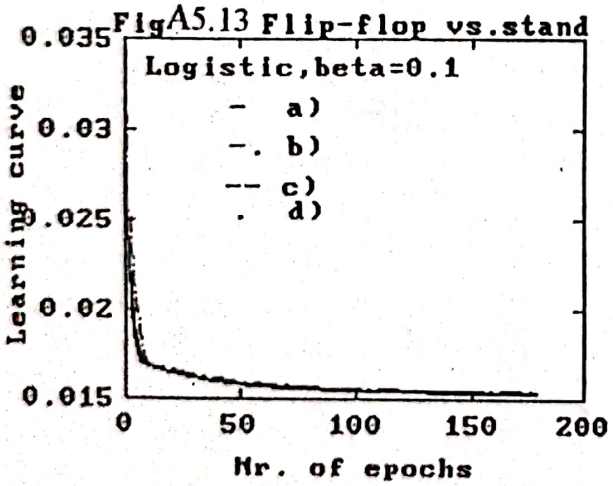


Fig. A5.13 Curbele de învăţare pentru strategiile "flip–flop" şi standard, funcţia cost logistică, $\beta = 0.1$, diferite valori ale parametrului γ şi funcţia de activare $f_2(s, net)$

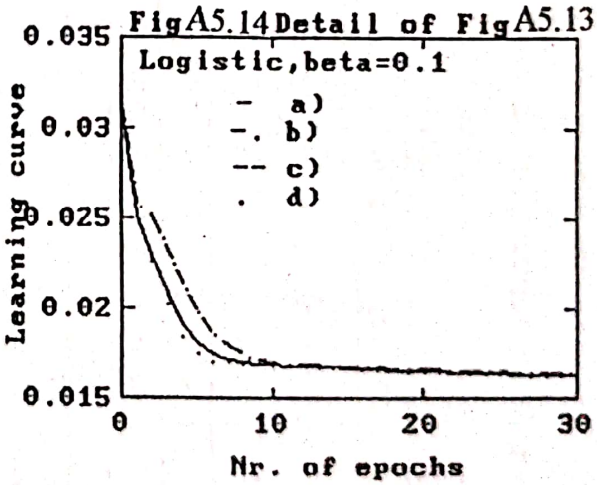


Fig. A5.14 Detaliu al figurii A5.13

Scanned with OKEN Scanner

Tab. A5.2 Eroarea de test în sens L_2, pentru $\gamma < 0$ şi $\gamma > 0$

Funcţia cost	Funcţia de activare	γ					Tendinţa erorii de test
		+0.00001	–0.00001	–0.0001	–0.001	–0.01	
Norma L_2	$f_1(s, net)$	0.064820	0.064614	0.064573	osc.*	osc.	↓
	$f_2(s, net)$	0.065335	0.065443	0.065701	osc.	osc.	↑
Hampel $\beta = 0.5$	$f_1(s, net)$	0.074197	0.072077	0.070824	osc.	osc.	↓
	$f_2(s, net)$	0.070422	0.070453	0.070624	osc.	osc.	↑
Logistică $\beta = 0.1$	$f_1(s, net)$	0.068096	0.068117	0.068271	0.068291	—	↑
	$f_2(s, net)$	0.068665	0.068680	0.068789	—	0.068873	↑

*osc. = oscilaţii

Tab. A5.3 Strategia de învăţare "flip – flop"

Funcţia cost	Funcţia de activare	$\lvert\gamma\rvert$	Eroare de test (L_2) după 90 de epoci, $\gamma < 0$	Eroarea de test (L_2) după alte 90 epoci, $\gamma > 0$	Eroarea de test (L_2), după 180 epoci, $\gamma > 0$
Norma L_2	$f_1(s, net)$	0.00001	0.064614	0.064835	0.064688
		0.00010	0.064573	0.064892	0.064601
	$f_2(s, net)$	0.00001	0.065443	0.064807	0.064823 *
		0.00010	0.065701	0.064776	0.064949 *
Logistică $\beta = 0.1$	$f_1(s, net)$	0.00010	0.068271	0.067512	0.067206
	$f_2(s, net)$	0.00001	0.068680	0.068107	0.068039
		0.00010	0.068789	0.068386	0.067797

Scanned with OKEN Scanner

ANEXA 6

INTERPRETAREA ÎN PLANUL COMPLEX A REZULTATELOR ÎNVĂȚĂRII REȚELEI NEURALE "FEEDFORWARD"

Interpretarea semnificaţiei valorilor ponderilor şi semnalelor de ieşire din fiecare strat, în procesul de învăţare al reţelelor neurale "feedforward" este o problemă dificilă, care se bazează uneori pe reprezentări grafice. Pentru valorile ponderilor conexiunilor, amintim diagramele Hinton şi diagramele de legătură ("bond diagrams") [H56].

Fig. A6.1 şi A6.2 conţin diagramele Hinton pentru ponderile conexiunilor reţelei neurale care a învăţat semnalul "chirp", conform precizărilor de la punctul 5.6.3. Dimensiunea pătratelor este proporţională cu valoarea ponderii conexiunii reprezentate. Dezavantajul diagramei este acela că, nu explicitează relaţia dintre topologia reţelei şi valorile ponderilor, astfel încât procesul de învăţare să poată fi urmărit, identificându-se ponderile care îşi modifică semnificativ valorile. Acest lucru se poate realiza cu ajutorul diagramelor de legătură, a căror utilizare măreşte însă considerabil complexitatea software-ului de simulare.

Având în vedere aceste observaţii, rezultă ideea reprezentării datelor din RNA în alte coordonate / plan, pentru înţelegerea procesului de învăţare. Posibilitatea stabilirii unei relaţii între transformările neliniare realizate în structura neurală "feedforward" şi transformările conforme în planul complex a fost studiată în lucrarea [D205]. Cele din urmă sunt utilizate în analiza complexă deoarece proiectează o regiune a planului într-o altă regiune, în care rezolvarea problemei de studiu poate fi realizată mai simplu.

Pentru interpretarea rezultatelor învăţării în planul complex, sunt necesare câteva elemente preliminare.

Dacă punctul de coordonate (α, β) din planul $\mathcal{R}^2 = \mathcal{R} \times \mathcal{R}$ se identifică numărului complex $z = \alpha + i\beta$, atunci $\mathcal{R}^2$ se numeşte planul complex $\boldsymbol{C}$ [K503].

Dacă z este o variabilă complexă, atunci $g(z)$ este o funcţie de variabilă complexă [J501].

Definiţia A6.1 Fie $g(z)$ o funcţie de variabilă complexă z, definită pe domeniul $D \subset \boldsymbol{C}$. Dacă $g(z)$ este o funcţie diferenţiabilă de z în sens complex,

astfel încât derivata $g'(z_0)$ este continuă adică, în fiecare punct $z_0 \in D$, este valabilă relaţia:

$$g'(z) = \lim_{z \to z_0} \frac{g(z) - g(z_0)}{z - z_0} \qquad \text{(A6.1)}$$

atunci funcţia $g(z)$ se numeşte funcţie olomorfă de z (sau "analitică", "complex diferenţiabilă") [K503, M209].

Propoziţia A6.1 Funcţia $g : \boldsymbol{C} \to \boldsymbol{C}$, $z \Rightarrow exp(z)$ este olomorfă în $\boldsymbol{C}$ şi derivata sa este egală cu $exp(z)$ [M209].

Definiţia A6.2 Transformările de forma (A6.2) se numesc transformări Möbius (sau transformări "omografice", "biliniare") [M208] şi sunt o clasă de transformări conforme.

$$\mathfrak{I}(z) = \frac{a\,z + b}{c\,z + d}, \text{ cu } a, b, c, d \in \boldsymbol{C} \text{ şi } ad - bc \neq 0 \qquad \text{(A6.2)}$$

Observaţia A6.1 Transformarea $\mathfrak{I}$ dată de relaţia (A6.3) nu determină în mod unic coeficienţii a, b, c, d. Dacă $\varphi \in \boldsymbol{C} \setminus \{0\}$ atunci, coeficienţii φa, φb, φc, φd corespund aceleiaşi transformări $\mathfrak{I}$ [J501].

Teorema A6.1 Fiecare transformare Möbius este compusă dintr-un număr finit de transformări Möbius de tipul [J501]:

- $\square$ $Rot(z) = e^{i\theta} z$ (rotaţia de unghi θ)
- $\square$ $Sim(z) = r\,z$ ($r > 0$) (transformare de similaritate, cu dilatarea / contracţia distanţelor cu factorul r)
- $\square$ $T_t(z) = z + t$ (translaţie)

Funcţia sigmoidală $f(z)$ de activare a nodurilor stratului ascuns din RNA "feedforward" ($z \in \boldsymbol{C}$), este o funcţie olomorfă. In lucrarea [D205] s-a arătat că, transformarea sigmoidală realizată de stratul ascuns al RNA "feedforward asupra semnalului complex $z = \alpha + i\,\beta$ este o transformare Möbius. Prin examinarea patternurilor obţinute la ieşirea stratului ascuns, reprezentate ca vectori, se poate identifica tipul transformării (rotaţie, dilatare / contracţie, translaţie, etc.) efectuate de nodul respectiv asupra patternurilor de intrare şi se poate explica rolul său în structura neurală, precum şi evoluţia sa în procesul de învăţare. De asemenea, s-a arătat că, în condiţii precizate, transformarea menţionată este o compunere de rotaţii elementare în planul complex.

Scanned with OKEN Scanner

Fie reţeaua neurală "feedforward" care a învăţat semnalul "chirp", conform precizărilor de la punctul 5.6.2.

In fig. A6.3 este reprezentat semnalul obţinut la ieşirea nodului ascuns $h = 1$, în comparaţie cu semnalul determinat prin efectuarea transformării de rotaţie în planul complex, asupra patternurilor de intrare complexe. Se observă că, în plus faţă de transformarea de rotaţie, intervine şi o scalare în amplitudine a semnalului.

Comparaţia realizată grafic în fig. A6.4, între semnalul obţinut la ieşirea reţelei şi semnalul determinat prin efectuarea transformării de rotaţie în planul complex, asupra patternurilor de intrare, arată că, transformarea intrare – ieşire realizată de reţea este în cazul problemei considerate, o compunere de rotaţii elementare [D205].

Reprezentarea sugestivă ca vectori, a patternurilor de intrare, a patternurilor de ieşire din stratul ascuns (fig. A6.5) şi detaliat, a patternurilor obţinute la ieşirea fiecărui nod ascuns (fig. A6.6), confirmă rezultatele amintite anterior.

Observaţiile realizate deschid noi perspective asupra problemei dificile a interpretării rezultatelor obţinute în procesul de învăţare, precum şi asupra problemei extragerii regulilor necesare unui sistem expert, din reţeaua neurală care a învăţat [D205].

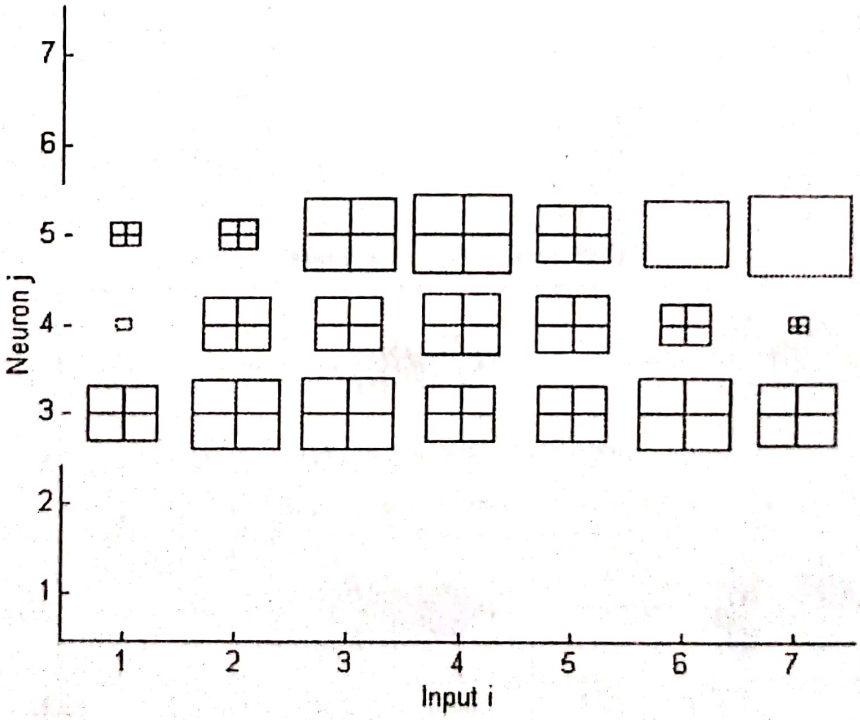


Fig. A6.1 Diagrama Hinton a ponderilor conexiunilor dintre stratul de intrare şi stratul ascuns, pentru RNA care a învăţat semnalul "chirp"

Scanned with OKEN Scanner

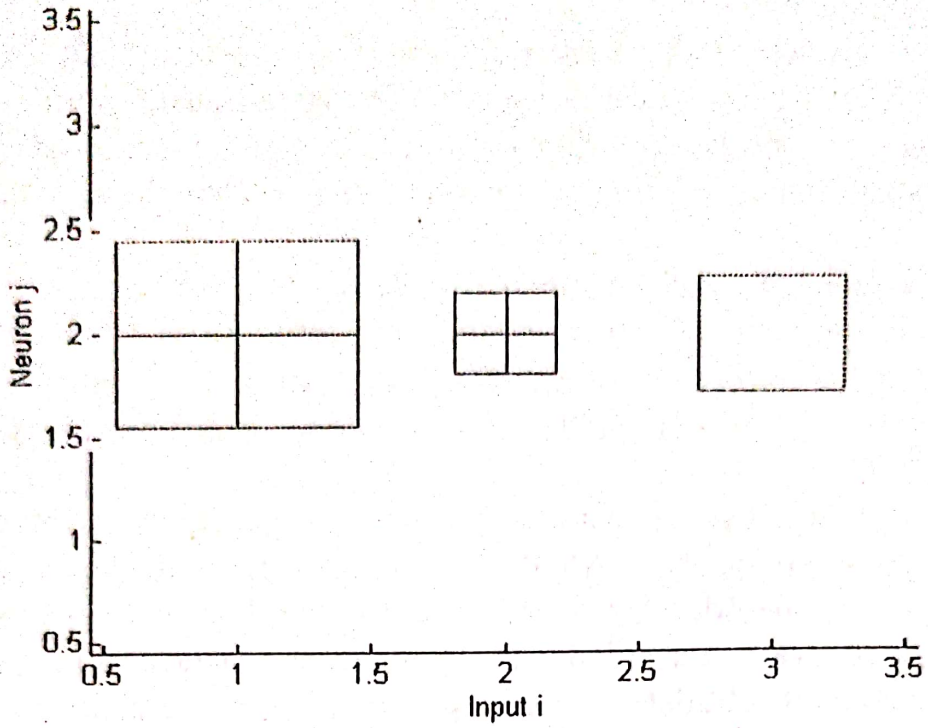


Fig. A6.2 Diagrama Hinton a ponderilor conexiunilor dintre stratul ascuns şi stratul de ieşire, pentru RNA care a învăţat semnalul "chirp"

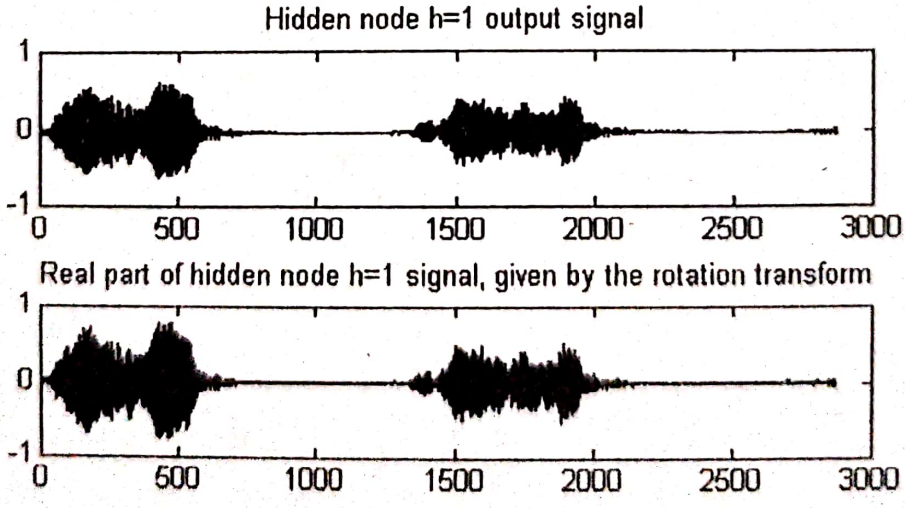


Fig. A6.3 Semnalul obţinut la ieşirea nodului ascuns $h = 1$ şi semnalul determinat pentru acelaşi nod, prin efectuarea rotaţiei, pentru RNA care a învăţat semnalul "chirp"

Scanned with OKEN Scanner

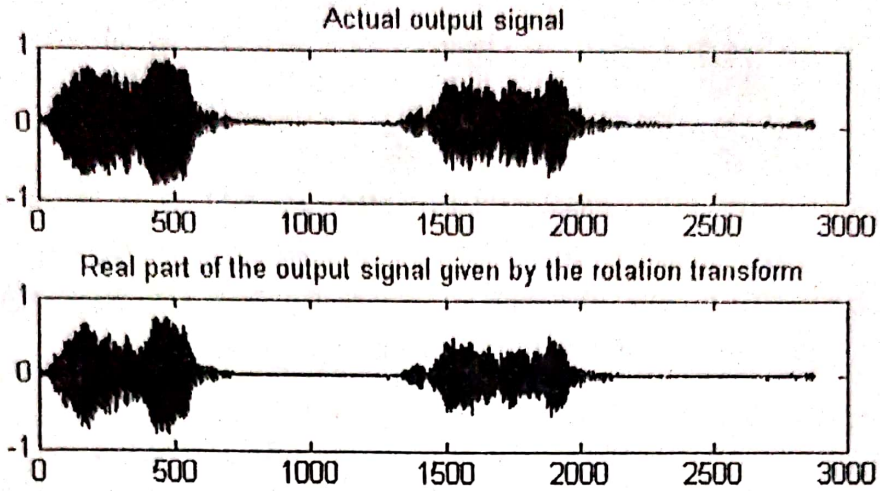


Fig. A6.4 Semnalul de ieşire obţinut şi semnalul de ieşire determinat prin efectuarea rotaţiei, pentru reţeaua care a învăţat semnalul "chirp"

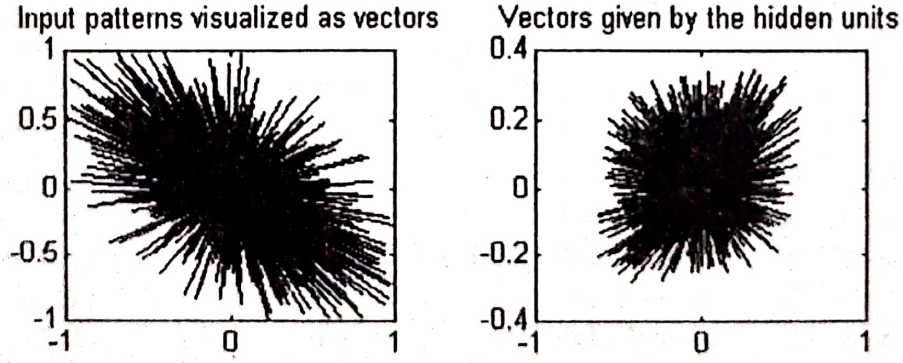


Fig. A6.5 Patternurile de intrare desenate ca vectori şi patternurile obţinute la ieşirile stratului ascuns. Se poate observa efectul global de rotaţie al transformării realizate de nodurile ascunse asupra patternurilor de intrare (cazul semnalului "chirp")

Scanned with OKEN Scanner

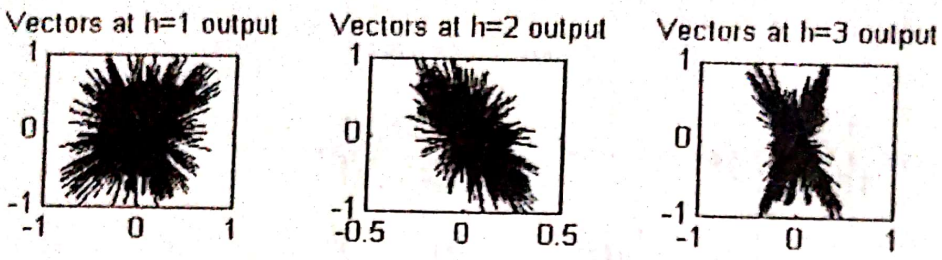


Fig. A6.6 Patternurile obţinute la ieşirea fiecărui nod din stratul ascuns, pentru RNA care a învăţat semnalul "chirp"

Scanned with OKEN Scanner

BIBLIOGRAFIE[24]

[A1] G. Avitable, M. Forti, S. Manetti, M. Marini – On a Class of Nonsymmetrical Neural Networks with Application to ADC; IEEE Trans. on Circuits and Systems, Vol. 38, No. 2, pp. 202–209, Febr. 1991.

[A4] Y.S. Abu–Mostafa, J.–M. St. Jacques – Information Capacity of the Hopfield Model; IEEE Trans. on Information Theory, Vol. 31, No. 4, pp. 461–464, Iulie 1985.

[A5] A. Adams, S.J. Bye – New Perceptron; Electronics Letters, Vol. 28, No. 3, pp. 321–322, SUA, Ian. 1992.

[A6] J. Austin – ADAM: An Associative Neural Architecture for Invariant Pattern Classification; IEE First Intl. Conference on Artificial Neural Networks, IEE Press, Conference Publication No. 313, pp. 196–200, 1989.

[A7] Y.S. Abu – Mostafa – Information Theory, Complexity and Neural Networks; IEEE Communications Magazine, Vol. 27, No. 11, pp. 25 – 81, Nov. 1989.

[A8] S. Amato, B. Apolloni, G. Caporali, ş.a. – Simulated Annealing Approach in Backpropagation; Neurocomputing, Vol. 3, No. 5 – 6, pp. 207 – 220, Elsevier, Olanda, Dec. 1991.

[A16] D. Anthony – The Use of Artificial Neural Networks in Medicine; preprint, accesibil prin ftp pe reţeaua Internet, 1991.

[A17] F. Albertini, E.D. Sontag – Uniqueness of Weights for Neural Networks; preprint, accesibil prin ftp pe reţeaua Internet.

[A22] B. Amirikian, H. Nishimura – What Size Network is Good for Generalization of a Specific Task of Interest ?; Neural Networks, Vol. 7, No. 2, pp. 321–329, Elsevier Science Ltd., SUA, 1994.

[A25] N.M. Allinson – Self–Organising Neural Maps and Their Applications; în J.G.Taylor, C.L.T. Mannion (eds.): Theory and Applications of Neural Networks; Springer – Verlag, pp. 101–118, Germania, 1992.

[A26] Shun–ichi Amari – Characteristics of Random Nets of Analog Neuron–Like Elements; în V. Vemuri (ed.): Neural Networks: Artificial Neural Networks: Theoretical Concepts; Computer Society Press Technology Series, The Computer Society of the IEEE, pp. 55–69, SUA, 1988.

[A27] N.M. Allinson, M.T. Brown, M.J. Johnson – $\{0, 1\}^n$ Space Self–Organising Feature Maps – Extensions and Hardware Implementations; Proc. of the First Intl. Conference on Artificial Neural Networks, IEE Press 1989, Conference Publication No. 313, pp. 261–264.

[A28] M.A. Arbib, Shun–ichi Amari (eds.) – Dynamic Interactions in Neural Networks; Springer Verlag, SUA, 1989.

[A29] J.B. Abbiss, B.J. Brames, M.A. Fiddy – Superresolution Algorithms for a Modified Hopfield Neural Network; IEEE Trans. on Neural Networks, Vol. 39, No. 7, pp. 1516–1523, Iulie 1991.

[24] 475 lucrări

Scanned with OKEN Scanner

[B19] E. Barnard, D. Cassasent – Invariance and Neural Nets; IEEE Trans. on Neural Networks, Vol. 2, No. 5, pp. 498–507, Sept. 1991.

[B23] E.B. Bartlett – Dynamic Node Architecture Learning: An Information Theoretic Approach; Neural Networks, Vol. 7, No. 1, pp. 129–140, Elsevier Science Ltd., SUA, 1994.

[B27] N.J. Bershad, J.J. Shynk – Performance Analysis of a Converged Single–Layer Perceptron for Nonseparable Data Models with Bias Term; IEEE Trans. on Signal Processing, Vol. 42, No. 1, Ian. 1994.

[B28] Y. Bengio, P. Simard, P. Frasconi – Learning Long – Term Dependencies with Gradient Descent is Difficult; IEEE Trans. on Neural Networks, Vol. 5, No. 2, pp. 157–166, Martie 1994.

[B29] M. Bianchini, M. Gori, M. Maggini – On the Problem of Local Minima in Recurrent Neural Networks; IEEE Trans. on Neural Networks, Vol. 5, No. 2, pp. 167–177, Martie 1994.

[B30] C.M. Bishop – Curvature – Driven Smoothing in Backpropagation Neural Networks; în J.G.Taylor, C.L.T. Mannion (eds.): Theory and Applications of Neural Networks; Springer –Verlag, pp. 139–148, Germania, 1992.

[B31] L. Bochereau, P. Bourgine – Rule Extraction and Validity Domain on a Multilayer Neural Network; Proc. of IJCNN'90: Intl. Joint Conference on Neural Networks, San Diego, California; IEEE, Intl., Vol. 1, pp. I 97 – I 100, SUA, 1990.

[B32] P.–Y. Burgi – Analysis and Understanding of Neuronal Structures; Neurocomputing, Vol. 4, No. 3–4, Elsevier Science Publ., pp. 137–165, Olanda, Mai 1992.

[B33] H. Bourlard, C. J. Wellekens – Links between Markov Models and Multilayer Perceptrons; IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 12, No. 12, pp. 1167–1178, Dec. 1990.

[B34] Y. Bengio, R. de Mori, G. Flammia, R. Kompe – Global Optimization of a Neural Network – Hidden Markov Model Hybrid; IEEE Trans. on Neural Networks, Vol. 3, No. 2, pp. 252–259, Martie 1992.

[B35] S. Bibyk, M. Ismail – Issues in Analog VLSI and MOS Techniques for Neural Computing; în C. Mead, M. Ismail (eds.): Analog VLSI Implementations of Neural Systems; Kluwer Academic Publ., pp. 103–133, SUA, 1989.

[B36] H.–U. Bauer,K.R. Pawelzik – Quantifying the Neighborhood Preservation of Self – Organizing Feature Maps; IEEE Trans. on Neural Networks, Vol. 3, No. 4, pp. 570–579, Iulie 1992.

[B37] Y. Bruck – On the Convergence Properties of the Hopfield Model; Proc. of the IEEE, Vol. 78, No. 10, pp. 1579–1585, Oct. 1990.

[B38] A. Björck – Error Analysis of Least Squares Algorithms; în G.H. Golub, P. Van Dooren (eds.): Numerical Linear Algebra, DSP and Parallel Algorithms, Springer Verlag, NATO ASI Series, Vol. F70, pp. 41 – 73, Germania, 1991.

[B39] M.G. Bellanger – The Family of Least Squares Algorithms for Adaptive Filtering; în G.H. Golub, P. Van Dooren (eds.): Numerical Linear Algebra, DSP and Parallel Algorithms, Springer Verlag, NATO ASI Series, Vol. F70, pp. 1–18, Germania, 1991.

[B40] H.–U. Bauer, T. Geisel – Sequence Analysis in Feedback Multilayer Perceptrons; în R. Eckmiller, G. Hartmann, G. Hauske (eds.): Parallel Processing in Neural Systems and Computers, Elsevier Science Publ., pp. 375–379, Olanda, 1990.

Scanned with OKEN Scanner

[C33] V. Cherkassky, K. Fasset, N. Vassilas – Linear Algebra Approach to Neural Associative Memories and Noise Performance of Neural Classifiers; IEEE Trans. on Computers, Vol. 40, No. 12, pp. 1429–1435, Dec. 1991.

[C34] T.–D. Chiueh, R.M. Goodman – The Kanerva Memory is Stable; Proc. of IJCNN'91: Intl. Joint Conference on Neural Networks, Seattle, Vol. 2, pp. II 267–II 271, SUA, 1991.

[C35] G.A. Carpenter, S. Grossberg – Adaptive Resonance Theory: Neural Network Architectures for Self – Organizing Pattern Recognition; în R. Eckmiller, G. Hartmann, G. Hauske (eds.): Parallel Processing in Neural Systems and Computers; Elsevier Science Publ., pp. 383–389, Olanda, 1990.

[C36] G.A. Carpenter, S. Grossberg, D. Rosen – ART2: An Adaptive Resonance Algorithm for Rapid Category Learning and Recognition; Proc. of IJCNN'91: Intl. Joint Conference on Neural Networks, Seattle, Vol. 2, pp. II 151–156, SUA, 1991.

[C37] M.A. Cohen, S. Grossberg – Absolute Stability of Global Pattern Formation and Parallel Memory Storage by Competitive Neural Networks; în V. Vemuri (ed.): Artificial Neural Networks: Theoretical Concepts; Computer Society Press Technology Series, The Computer Society of the IEEE, pp 70–81, SUA, 1988.

[C38] K. Chakraborty, K. Mehrotra, C.K. Mohan, S. Ranka – Forecasting the Behavior of Multivariate Time Series Using Neural Networks; Neural Networks, The Pergamon Press, Vol. 5, No. 6, pp. 961–970, SUA, 1992.

[C39] P. Cardaliaguet – Aproximation of a Function and its Derivative with a Neural Network; Neural Networks, The Pergamon Press, Vol. 5, No. 2, pp. 207–220, SUA, 1991.

[C40] S. Carrato, S. Marse – Parallel Structure Based on Neural Networks for Image Compression; Electronics Letters, Vol. 28, No. 12, pp. 1152–1153, Iulie 1992.

[C41] G. Chinn, S.–C. Huang – Weighted Least – Squares Filtered Backprojection Tomographic Reconstruction; IEEE Signal Processing Letters, Vol. 2, No. 3, pp. 49–50, Martie 1995.

[C42] N.G. Chernoguz – A Smoothed Newton – Gauss Method with Application to Bearing – Only Position Location; IEEE Trans. on Signal Processing, Vol. 43, No. 8, pp. 2011–2013, Aug. 1995.

[C43] D.S. Chen, R.C. Jain – A Robust Back Propagation Learning Algorithm for Function Approximation; IEEE Trans. on Neural Networks, Vol. 5, No. 3, pp. 467–479, Mai 1994.

[C44] A. Cichocki, R. Unbehauen – Simplified Neural Network for Solving Linear Least Squares and Total Least Squares Problems in Real Time; IEEE Trans. on Neural Networks, Vol. 5, No. 6, pp. 910–923, Nov. 1994.

[C46] N.S. Cardell, W.H. Joerding, Y. Li – Symmetry Constraints for Feedforward Network Models of Gradient Systems; IEEE Trans. on Neural Networks, Vol. 6, No. 5, pp. 1249–1254, Sept. 1995.

[C47] L.– H. Chen, S. Chang – An Adaptive Learning Algorithm for Principal Component Analysis; IEEE Trans. on Neural Networks, Vol. 6, No. 5, pp. 1255–1263, Sept. 1995.

[C500] Y. Le Cun, J. Denker, S. Solla – Optimal Brain Damage; în D. Touretzky (ed.): Advances in Neural Information Processing Systems, Vol. 2, San Mateo, Morgan Kaufmann, SUA, 1989.

Scanned with OKEN Scanner

[D200] J.S. Denker, Y. Le Cun, S.A. Solla – Optimal Brain Damage; în D. Touretzky (ed.): Advances in Neural Information Processing Systems, Morgan Kaufmann, San Mateo, Ca, Vol. 2, SUA, 1990.

[D201] Adriana Dumitraş, A.T. Murgan, V. Lăzărescu – A Quantitative Study of Evoked Potential Estimation Using a Feedforward Neural Network; Proc. of the 1994 IEEE Workshop on Neural Networks for Signal Processing, Ermioni, IEEE; J. Vlontzos, J.–N. Hwang, E. Wilson (eds), pp. 606–615, Grecia, Sept. 1994.

[D202] Adriana Dumitraş, V. Lăzărescu, A.T. Murgan – A Growing – Decreasing Method for Designing Neural Filters; Proc. of the IEEE Workshop on Nonlinear Signal and Image Processing, Neos Marmaras, Halkidiki; I. Pitas (ed.), Vol. 2, pp. 579–582, Grecia, Iunie 1995.

[D203] Adriana Dumitraş, E. Rădulescu, V. Lăzărescu – Improved Classification of Psychiatric Mood Disorders Using a Feedforward Neural Network; Proc. of MEDINFO'95: 8th World Congress on Medical Informatics, Vancouver; R.A. Greenes, H.E. Peterson, D.J. Protti (eds.), Vol. 1, pp. 818–822, IMIA, Canada, Iulie 1995.

[D204] Adriana Dumitraş, V. Lăzărescu, A.T. Murgan – On Designing a Neural Filter with Adaptive Slope of the Node's Activation Function; Proc. of ECCTD'95: The 12th European Conference on Circuit Theory and Design, Vol. 1, pp. 195–198, ISBN 975–561–061–8, Istanbul, Turcia, Aug. 1995.

[D205] Adriana Dumitraş – Structuri de rețele neurale artificiale pentru prelucrarea digitală a semnalelor; Teză de doctorat, Facultatea Electronică şi Telecomunicaţii, Universitatea "Politehnica" Bucureşti, 1996.

[D206] Adriana Dumitraş, V. Lăzărescu – A "Flip – Flop" Learning Strategy in Feedforward Neural Networks; Romanian Journal of Biophysics, Ed. Academiei, Vol. 5, No. 1, pp. 9–28, Bucureşti, România, Ian.–Martie 1995.

[D207] Adriana Dumitraş, V. Lăzărescu – MOBD: A Neural Network Pruning Algorithm with Symmetry Constraints; Proc. of ICSP'96: Intl. Conference on Signal Processing, Beijing, China, Oct. 1996, în curs de publicare.

[D208] Adriana Dumitraş, V. Lăzărescu – The Influence of the MLP's Output Dimension on Its Performance in Image Restoration; Proc. of ISCAS'96: The IEEE Intl. Symposium on Circuits and Systems, Vol. 1, pp. 329–332, ISBN 0–7803–3073–0, Atlanta, Georgia, SUA, Mai 1996.

[D209] Adriana Dumitraş, V. Lăzărescu – A Feedforward Neural Network vs. An Expert System for E.E.N. (T.) Classification; Proc. of the Third World Congress on Expert Systems, Seoul; J.K.Lee, J. Liebowitz, Y.M. Chae (eds.): Cognizant Communication Corporation, ISBN 1–882345–04–5, Vol. 2, pp. 962 – 969, Coreea, 1996.

[D210] Adriana Dumitraş, V. Lăzărescu – Neural Network Structures Simulated with MatLab. A Labwork Software Package for Undergraduate Students; Proc. of EUFIT'96: The Fourth European Congress on Intelligent Techniques and Soft Computing, Aachen; Verlag Mainz, Wissenschafts Verlag, Aachen, ISBN 3–89653–187–5, Vol. 3, pp. 2269–2273, Germania, 1996.

[D211] A. Doering, M. Galicki, H. Witte, V. Krajca – Structure Optimization of Neural Networks with A* Algorithm. Application in EEG Pattern Analysis; Proc. of MEDINFO'95: 8th World Congress on Medical Informatics, Vancouver; R.A. Greenes, H.E. Peterson, D.J. Protti (eds.), Vol. 1, pp. 814–817, IMIA, Canada, 1995; ISBN 0–9697414–1–3.

Scanned with OKEN Scanner

[F36] K. Fukushima – Character Recognition with Neural Networks; Neurocomputing, Elsevier Science Publ., Vol. 4, No. 5, pp. 221–233, Aug. 1992.

[F37] E. Fiesler – Neural Network Classification and Formalization; preprint, accesibil prin ftp pe rețeaua Internet; acceptat spre publicare în Computer Standards & Interfaces, J. Fulcher (ed.), Vol. 16, Elsevier Science Publ., Olanda, 1994.

[F500] B. Fritzke – Unsupervised Clustering with Growing Cell Structures; Proc. of IJCNN'91: Intl. Joint Conference on Neural Networks, Seattle, Vol. 2, pp. 531–536, SUA, 1991.

[F501] B. Fritzke – Growing Cell Structures – A Self–Organizing Network for Unsupervised and Supervised Learning; Neural Networks, Vol. 7, No. 9, pp. 1441–1460, Elsevier Science Ltd., SUA, 1994.

[F502] M. Frean – The Upstart Algorithm: A Method for Constructing and Training Feedforward Neural Networks; Neural Computation, No. 2, pp. 198–209, SUA, 1990.

[G7] A.H. Gee, R.W. Prager – Alternative Energy Functions for Optimizing Neural Networks; Report, Cambridge University, Engineering Dept., CUED/F – INFENG/TR 95, Cambridge, Anglia, Martie 1992.

[G8] N. Graham – Learning C++; Osborne Mc–Graw Hill, Inc., SUA, 1991.

[G11] A.S. Gevins, N.H. Morgan – Applications of Neural Network Signal Processing in Brain Research; IEEE Trans. on Acoustics, Speech and Signal Processing, Vol. 36, No. 7, pp. 1152–1161, Iulie 1988.

[G12] U. Guyon, U. Poujaud, L. Personnaz, G. Dreyfus – Comparing Different Neural Network Architectures for Classifying Handwritten Digits; Proc. of IJCNN'89: Intl. Joint Conference on Neural Networks, Vol. 2, pp. II 127–II 132.

[G15] J. Gorodkin, L.K. Hansen, A. Krogh, C. Svarer, O.Winther – A Quantitative Study of Pruning by Optimal Brain Damage; preprint, Electronics Institute, Technical University of Denmark, Ian. 1993; accesibil prin ftp pe rețeaua Internet.

[G16] A.H. Gee, R.W. Prager – Alternative Energy Functions for Optimizing Neural Networks; Technical Report CUED/F–INFENG/TR 95, Cambridge University Engineering Department, England, March 1992; preprint, accesibil prin ftp pe rețeaua Internet.

[G17] H. Gish – Maximum Likelihood Training of Neural Networks; în D.J. Hand (ed.): Artificial Intelligence Frontiers in Statistics; Chapman and Hall, pp. 241–255, UK, 1993.

[G19] R.J. Gaynier, T. Downs – On the Properties of Error Functions that Affect the Speed of Backpropagation Learning; Proc. of ICANN'94: Intl. Conference on Artificial Neural Networks, Sorrento, pp. 557–560, Italia, 1994.

[G21] C.L. Giles, C.W. Omlin – Pruning Recurrent Neural Networks for Improved Generalization Performance; IEEE Trans. on Neural Networks, Vol. 5, No. 5, pp. 848–851, Sept. 1994.

[G22] V.K. Govindan – Character Recognition: A Review; Pattern Recognition, The Pergamon Press, Vol. 23, No. 7, pp. 671–683, UK, 1990.

[G23] G.H. Golub, P. Van Dooren (eds.) – Numerical Linear Algebra, DSP and Parallel Algorithms; Springer Verlag, NATO ASI Series, Vol. F70, Germania, 1991.

[G24] A.A. Ghorbani, V.C. Bhavsar – Incremental Communication for Multilayer Neural Networks; IEEE Trans. on Neural Networks, Vol. 6, No. 6, pp. 1375–1385, Nov. 1995.

Scanned with OKEN Scanner

[H30] Y.H. Hu – Configuration of Feedforward Multilayer Perceptron Neural Networks; preprint, University of Wisconsin – Madison, Dept. of Electrical and Computer Engineering, Nov. 1993.

[H32] Yu Hen Hu – Special Topics in Applications of Neural Networks to Signal Processing; Lecture Notes, University of Wisconsin – Madison, Dept. of Electrical and Computer Engineering, Aug. 1994.

[H38] B.K. Humpert – Improving Back Propagation with a New Error Function; Neural Networks, Vol. 7, No. 8, Elsevier Science Ltd., SUA, 1994, pp. 1191–1192.

[H39] K. Hagiwara, S. Usui – Numerical Experiments on the Information Criteria for Layered Feedforward Neural Nets; Proc. of ICANN'94: Intl. Conference on Artificial Neural Networks, Sorrento, pp. 493–496, Italia, 1994.

[H41] S. Haykin – Neural Networks. A Comprehensive Foundation; MacMillan College Publ. Co., SUA, 1994.

[H42] R. Hecht – Nielsen – Towards Hierarchical Matched Filtering; în R. J. Mammone, Yehoshua Zeevi (eds.): Neural Networks. Theory and Applications; Academic Press Inc., Harcourt Brace Jovanovitch Publ., pp. 217–232, SUA, 1991.

[H43] Y. Hussain, N. Farvardin – Adaptive Block Transform Coding of Speech Based on LPC Vector Quantization; IEEE Trans. on Signal Processing, Vol. 39, No. 12, pp. 2611–2620, Dec. 1991.

[H44] Y. Hirai – Hardware Implementation of Neural Networks in Japan; Neurocomputing, Elsevier Science Publ., Vol. 5, No. 1, pp. 3–16, Olanda, Febr. 1993.

[H46] L. Holmström, P. Koistinen – Using Additive Noise in Back Propagation Training; IEEE Trans. on Neural Networks, Vol. 3, No. 1, pp. 24–37, Ian. 1992.

[H47] M. Heywood, P. Noakes – A Framework for Improved Training of Sigma–Pi Networks; IEEE Trans. on Neural Networks, Vol. 6, No. 4, pp. 893–903, Iulie 1995.

[H48] R. Harvey, J.A. Bangham, R.V. Aldridge, G.D. Tattersal – Information Reduction by Linear and Nonlinear Digital Filters; Proc. of the IEEE Workshop on Nonlinear Signal and Image Processing; Ioannis Pitas (ed.), Vol. 2, pp. 1023 – 1025, Neos Marmaras, Grecia, 1995.

[H50] J.B. Hampshire II, B.A. Pearlmutter – Equivalence Proofs for Multilayer Perceptron Classifiers and the Bayesian Discriminant Function; Proc. of the 1990 Connectionist Models Summer School, D. Touretzky, Elman, T. Sejnowski, G. Hinton (eds.), San Mateo, Morgan Kaufmann, SUA, 1990.

[H53] L.K. Hansen – Boltzmann Learning of Parameters in Bayes Visual Reconstruction; Proc. of the First Danish Conference on Pattern Recognition and Machine Intelligence, Dept. of Computer Science, University of Copenhagen; S.J. Olsen (ed.), Danemarca, 1992.

[H54] L.K. Hansen – Boltzmann Learning of Parameters in Celullar Neural Networks; Proc. of CNNA'92: Intl. Conference on Celullar Neural Networks and Their Applications, Munchen; J.A. Nossek (ed.), pp. 62 – 67, Germania, 1992.

[H56] S. Haykin – Neural Networks Expand Signal Processing's Horizons; IEEE Signal Processing Magazine, pp. 24–49, Martie 1996.

[H58] D.M. Hummels, W. Ahmed, M.T. Musavi – Adaptive Detection of Small Sinusoidal Signals in Non – Gaussian Noise Using and RBF Neural Network; IEEE Trans. on Neural Networks, Vol. 6, No. 1, pp. 214–219, Ian. 1995.

Scanned with OKEN Scanner

[J16] B. Jeong, Y.H. Lee – Design of Weighted Order Statistic Filters Using the Perceptron Algorithm; IEEE Trans. on Signal Processing, Vol. 42, No. 11, Nov. 1994.

[J17] R.C. Johnson – Conference Report about IJCNN'91: Intl. Joint Conf. on Neural Networks, Seattle, SUA; Neurocomputing, Vol. 3, No. 4, Elsevier Science Publ., pp. 193–196, Olanda, 1991.

[J18] B.H. Jansen – Time Series Analysis by Means of Linear Modelling; în R. Weitkunat (ed.): Digital Biosignal Processing; Elsevier Science Publ., pp. 157–179, Olanda, 1991.

[J19] L.K. Jones – Constructive Approximations for Neural Networks by Sigmoidal Functions; Proc. of the IEEE, Vol. 78, No. 10, pp. 1586–1589, Oct. 1990.

[J20] C. Ji, D. Psaltis – The Capacity of a Two Layer Network with Binary Weights; Proc. of IJCNN'91: Intl. Joint Conference on Neural Networks, Seattle, Vol. 2, pp. II. 127–II 137, SUA, 1991.

[J200] A.K. Jain – Fundamentals of Digital Image Processing; Englewood Cliffs, NJ, Prentice – Hall, 1990.

[J500] C. Ji, R.R. Snapp, D. Psaltis – Generalizing Smoothness Constraints From Discrete Samples; Neural Computation, Vol. 2, No. 2, pp. 188–197, 1990.

[J501] G.A. Jones, D. Singerman – Complex Functions; Cambridge Univ. Press, UK, 1991.

[K2] T. Khana – Foundations of Neural Networks; Addison – Wesley Publ. Co., 1990.

[K3] T. Kohonen – Associative Memory. A System – Theoretical Approach; Springer – Verlag, Germania, 1978.

[K7] B. Kosko – Neural Networks and Fuzzy Systems. A Dynamical Systems Approach to Machine Intelligence; Prentice – Hall Intl., Inc., SUA, 1992.

[K8] B.J. Kröse, P. Patrick van der Smagt – An Introduction to Neural Networks; University of Amsterdam, Faculty of Mathematics and Computer Science, The Netherlands, Ian. 1993; accesibil prin ftp pe reţeaua Internet.

[K9] B. Kosko (ed.) – Neural Networks for Signal Processing; Prentice – Hall Intl., Inc., 1993.

[K11] A. Kelley, I. Pohl – An Introduction to Programming in C; The Benjamin/ Cummings Publ. Co., Inc., SUA, 1984.

[K19] A. Krogh, J.A. Hertz - A Simple Weight Decay Can Improve Generalization; preprint, Electronics Institute, Technical University of Denmark, 1993; accesibil prin ftp pe reţeaua Internet.

[K23] B.J. Kagle, J.H. Murphy, ş.a. – Multi – Fault Diagnosis of Electronic Circuit Boards Using Neural Networks; Proc. of IJCNN'90: Intl. Joint Conference on Neural Networks, San Diego, Vol. 2, pp. II 197–II 202, SUA, 1990.

[K27] S.Y. Kung – Digital Neural Networks; PTR Prentice – Hall Inc., Englewood Cliffs, NJ, SUA, 1993.

[K29] R. Kane, M. Milgram – Forecasting Using Constrained Neural Networks; Proc. of ICANN'94: Intl. Conference on Artificial Neural Networks, Sorrento, pp. 284–287, Italia, 1994.

[K30] T. Kohonen – What Generalizations of the Self–Organizing Map Make Sense ?; Proc. of ICANN'94: Intl. Conference on Artificial Neural Networks, Sorrento, pp. 292–297, Italia, 1994.

[K32] R. Kamimura, S. Nakanishi – Information Minimization to Improve Generalization Performance; Proc. of ICANN'94: Intl. Conference on Artificial Neural Networks, Sorrento, pp. 717–720, Italia, 1994.

Scanned with OKEN Scanner

[M209] J.E. Marsden, M.J. Hoffman – Basic Complex Analysis; W.E. Freeman & Co., SUA, 1987.

[M500] M. Mézard, J.P. Nadal – Learning in Feedforward Layered Networks: The Tiling Algorithm; Journal of Physics, No. 22, pp. 2191–2203, SUA, 1989.

[M501] G. Mirchandani, W. Cao – On Hidden Nodes for Neural Nets; IEEE Trans. on Circuits and Systems, Vol. 36, No. 5, pp. 661–664, Mai 1989.

[M502] M.C. Mozer, P. Smolensky – Skeletonization: A Technique for Trimming the Fat from a Network via Relevance Assessment; în D. Touretzky (ed.): Advances in Neural Information Processing Systems, Morgan Kaufmann, San Mateo, CA, Vol. 1, SUA, 1989.

[M503] J. Moody, C.J. Darken – Fast Learning in Neural Networks of Locally Tuned Processing Units; Neural Computation, No. 1, pp. 281–294, 1989.

[N3] G. Nagy – Neural Networks – Then and Now; IEEE Trans. on Neural Networks, Vol. 2, No. 2, pp. 316–318, Martie 1991.

[N6] D.P. Morgan, C.L. Scofield – Neural Networks and Speech Processing; Kluwer Academic Publ., SUA, 1991.

[N7] B. Nabet, R.B. Pinter – Sensory Neural Networks: Lateral Inhibition; CRC Press, SUA, 1991.

[N9] D. Nguyen, B. Widrow – The Truck Backer – Upper: An Example of Self– Training in Neural Networks; Proc. of IJCNN'89: Intl. Conference on Neural Networks, Vol. 2, pp. II 357– II 363, SUA, 1989.

[N12] T.M. Nahban, A.Y. Zomaya – Toward Generating Neural Network Structures for Function Approximation; Neural Networks, Vol. 7, No. 1, pp. 89–99, Elsevier Science Ltd., SUA, 1994.

[N13] R. Neuneier, F. Hergert, W. Finnoff, D. Ormoneit – Estimation of Conditional Densities: A Comparison of Neural Network Approaches; Proc. of ICANN'94: Intl. Conference on Artificial Neural Networks, Sorrento, pp. 689–692, Italia, 1994.

[N15] O. Nerrand, P. Roussel–Ragot, D. Urbani, L. Personnaz, G. Dreyfus – Training Recurrent Neural Networks: Why and How ? An Illustration in Dynamical Process Modeling; IEEE Trans. on Neural Networks, Vol. 5, No. 2, pp. 178–184, Martie 1994.

[N16] M. Nathan, S.E. Tavares – Petri Nets for Classification; Electronics Letters, Vol. 28, No. 10, pp. 965–966, Mai 1992.

[N17] S. Niu, D.G. Fisher – Simultaneous Estimation of Process Parameters, Noise Variance and Signal – To – Noise Ratio; IEEE Trans. on Signal Processing, Vol 43, No. 7, pp. 1725– 1728, Iulie 1995.

[N18] V. Nedeljković – A Novel Multilayer Neural Networks Training Algorithm that Minimizes the Probability of Classification Error; IEEE Trans. on Neural Networks, Vol. 4, No. 4, pp. 650–659, Iulie 1993.

[N20] V. Neagoe, O. Stănăşilă – Teoria Recunoaşterii Formelor; Ed. Academiei, Bucureşti, România, 1992.

[O2] S. Olafsson – Some Dynamical Properties of Neural Networks; în R. Linggard, D.J. Myers, C. Nightingale (eds.): Neural Networks for Vision, Speech and Natural Language; Chapman & Hall, pp. 410–438, UK, 1992.

[O5] D. Obradovic – Recursive Learning in Recurrent Neural Networks with Varying Architecture; Proc. of ICANN'94: Intl. Conference on Artificial Neural Networks, Sorrento, pp. 447–450, Italia, 1994.

Scanned with OKEN Scanner

[S45] S. Schwartz, J. Wiles, I. Gough, S. Phillips – Connectionist, Rule - Based and Bayesian Decision Aids: An Empirical Comparison; în D.J.Hand (ed.): Artificial Intelligence Frontiers in Statistics; Chapman and Hall, pp. 264–277, Anglia, 1993.

[S46] D.F. Specht – A General Regression Neural Network; IEEE Trans. on Neural Networks, Vol. 2, No. 6, pp. 568 –575, Nov. 1991.

[S47] E. Säckinger, B.E. Boser, ş.a. – Application of the ANNA Neural Network Chip to High Speed Character Recognition; IEEE Trans. on Neural Networks, Vol. 3, No. 3, pp. 498–505, Mai 1992.

[S50] P. Patrick Van Der Smagt – Minimisation Methods for Training Feedforward Neural Networks; Neural Networks, Elsevier Science Publ., Vol. 7, No. 1, pp. 1–11, Olanda, 1994.

[S54] R. Schandry – Origins of the Main Human Biosignals; în R. Weitkunat (ed.): Digital Biosignal Processing; Elsevier Science Publ., pp. 1–12, Olanda, 1991.

[S55] R.H. Stottler, A.L. Henke – Automatic Translation from an Expert System to a Neural Network Representation; Proc. of the IJCNN'92: Intl. Joint Conference on Neural Networks, Baltimore, SUA, Vol. 1, pp. I 13–I 20, SUA, 1992.

[S56] S. Shinomoto – A Memory with Cognitive Ability; în M.A. Arbib, Shun–ichi Amari (eds.): Dynamic Interactions in Neural Networks; Springer Verlag, SUA, 1989, pp. 73–85, SUA, 1989.

[S57] C.–H. Sung – Percognitron: Neocognitron Coupled with Perceptron; Proc. of IJCNN'90: Intl. Joint Conference on Neural Networks, San Diego, CA, Vol. 3, pp. II 753–II 758, SUA, 1990.

[S59] D.G. Stork, G. Wolff, E. Levine – Neural Network Lipreading System for Improved Speech Recognition; Proc. of IJCNN'92: Intl. Joint Conference on Neural Networks, Baltimore, Vol. 2, pp. II 286–II 295, SUA, 1992.

[S61] K. Sivakumar, U.S. Desai – Image Restoration Using a Multilayer Perceptron with a Multilevel Sigmoidal Function; IEEE Trans. on Signal Processing, Vol. 41, No. 5, pp. 2018–2022, Mai 1993.

[S62] D.F. Stubbs – Three Applications of Neurocomputing in Biomedical Research; Neurocomputing, Elsevier Science Publ., The Netherlands, Vol. 2, No. 2, pp. 61–65, Nov. 1990.

[S63] D.T.M. Stock, T. Kailath – Numerically Stable Fast Transversal Filters for Recursive Least – Squares Adaptive Filtering; în G.H. Golub, P. Van Dooren (eds.): Numerical Linear Algebra, PDS and Parallel Algorithms; Springer Verlag, NATO ASI Series, Vol. F70, pp. 605–615, Germania, 1991.

[S64] K.M. Sammut, S.R. Jones – Implementing Nonlinear Activation Functions in Neural Network Emulators; Electronics Letters, Vo. 27, No. 12, pp. 1037–1038, Iunie 1992.

[S65] J.J. Shynk, S. Roy – Convergence Properties and Stationary Points of a Perceptron Learning Algorithm; Proc. of the IEEE, Vol. 78, No. 10, pp. 1599–1603, Oct. 1990.

[S66] J. Si, A.N. Michel – Analysis and Synthesis of a Class of Discrete – Time Neural Networks with Multilevel Threshold Neurons; IEEE Trans. on Neural Networks, Vol. 6, No. 1, pp. 105–116, Ian. 1995.

[S67] Y. Shin, J. Ghosh – Ridge Polynomial Networks; IEEE Trans. on Neural Networks; Vol. 6, No. 3, pp. 610–622, Mai 1995.

[S68] C. Svarer, L.K.Hansen, J. Larsen, C.E. Rasmussen – Designer Networks for Time Series Processing; Proc. of the 1993 IEEE Workshop on Neural Networks for Signal Processing, Baltimore, eds.: C.A. Kamm ş.a., SUA, 1993.

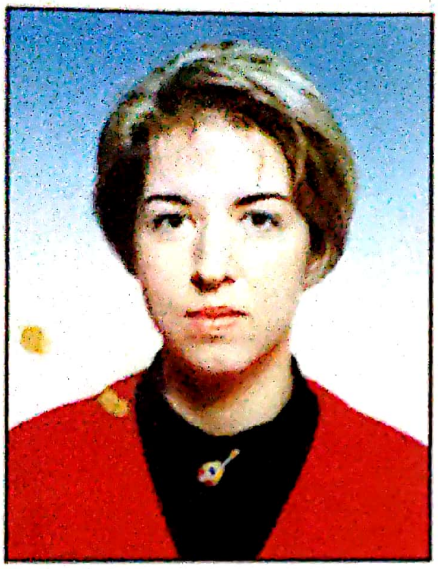
Scanned with OKEN Scanner

Editori fundatori:
NICOLAE CACOVEAN ȘI CONSTANTIN VOICULESCU

Format 16/70x100, cole tipo: 25,75, hârtie Scris I.A.
Tiraj: 500 exemplare, broșat.

Tiparul executat la INFOPRESS S.A.
sub comanda nr. 30/b.
1997

Scanned with OKEN Scanner



Autoarea s-a născut în București, la data de 24 martie 1965. A absolvit în anul 1988 Facultatea Electronică și Telecomunicații, Universitatea POLITEHNICA București, iar în 1996 a obținut titlul de doctor în specialitatea Electronică. În prezent, funcționează ca șef de lucrări la aceași facultate, catedra Electronică Aplicată.

Este co-autoare la o carte și peste 25 de lucrări publicate în reviste științifice și proceedings ale conferințelor din țară și străinătate. A participat în persoană la 5 conferințe internaționale (The 1994 IEEE Workshop on Neural Networks for Signal Processing, Ermioni, Grecia, 1994; The IEEE Workshop on Nonlinear Signal and Image Processing, Neos Marmaras, Grecia, 1995; The 8th World Congress on Medical Informatics, Vancouver, British Columbia, Canada, 1995; The 12th European Conference on Circuit Theory and Design, Istanbul, Turcia, 1995; The 4th European Conference on Intelligent Techniques and Soft Computing, Aachen, Germania, 1996).

În anul 1993 a fost bursieră la Universitatea Tehnică din Danemarca, Institutul de Teoria Circuitelor și Telecomunicații.

Principalele sale domenii de interes sunt: rețelele neurale artificiale și prelucrarea digitală a semnalelor.

* * *

Rețelele neurale artificiale reprezintă obiectul de studiu al domeniului interdisciplinar, aflat la granița dintre inginerie, științele cunoașterii, știința calculatoarelor și științele fundamentale: matematica, fizica, biologia etc. Apariția și dezvoltarea acestuia a fost rezultatul interferenței ideilor preluate din domeniile menționate, precum și al interesului cercetătorilor cu preocupări diverse. În acest moment, există o comunitate științifică internațională bine conturată, cu o activitate remarcabilă în domeniul rețelelor neurale.

În țara noastră, deși în ultimii ani au existat preocupări pentru studiul rețelelor neurale artificiale, mai ales în mediul universitar, acest interes a fost susținut de rare apariții editoriale. De aceea, în peisajul livresc actual, sperăm că lucrarea de față va contribui, la stimularea interesului și la promovarea rezultatelor cercetării în domeniul menționat.

Lei 25.000

ISBN 973-9008-75-5

Scanned with OKEN Scanner

Autoarea s-a născut în București, la data de 24 martie 1965. A absolvit în anul 1988 Facultatea Electronică și Telecomunicații, Universitatea POLITEHNICA București, iar în 1996 a obținut titlul de doctor în specialitatea Electronică. În prezent, funcționează ca șef de lucrări la aceași facultate, catedra Electronică Aplicată.

Este co-autoare la o carte și peste 25 de lucrări publicate în reviste științifice și proceedings ale conferințelor din țară și străinătate. A participat în persoană la 5 conferințe internaționale (The 1994 IEEE Workshop on Neural Networks for Signal Processing, Ermioni, Grecia, 1994; The IEEE Workshop on Nonlinear Signal and Image Processing, Neos Marmaras, Grecia, 1995; The 8th World Congress on Medical Informatics, Vancouver, British Columbia, Canada, 1995; The 12th European Conference on Circuit Theory and Design, Istanbul, Turcia, 1995; The 4th European Conference on Intelligent Techniques and Soft Computing, Aachen, Germania, 1996).

În anul 1993 a fost bursieră la Universitatea Tehnică din Danemarca, Institutul de Teoria Circuitelor și Telecomunicații.

Principalele sale domenii de interes sunt: rețelele neurale artificiale și prelucrarea digitală a semnalelor.

* * *

Rețelele neurale artificiale reprezintă obiectul de studiu al domeniului interdisciplinar, aflat la granița dintre inginerie, științele cunoașterii, știința calculatoarelor și științele fundamentale: matematica, fizica, biologia etc. Apariția și dezvoltarea acestuia a fost rezultatul interferenței ideilor preluate din domeniile menționate, precum și al interesului cercetătorilor cu preocupări diverse. În acest moment, există o comunitate științifică internațională bine conturată, cu o activitate remarcabilă în domeniul rețelelor neurale.

În țara noastră, deși în ultimii ani au existat preocupări pentru studiul rețelelor neurale artificiale, mai ales în mediul universitar, acest interes a fost susținut de rare apariții editoriale. De aceea, în peisajul livresc actual, sperăm că lucrarea de față va contribui, la stimularea interesului și la promovarea rezultatelor cercetării în domeniul menționat.

lei 25.000

ISBN 973-9008-75-5

Scanned with OKEN Scanner